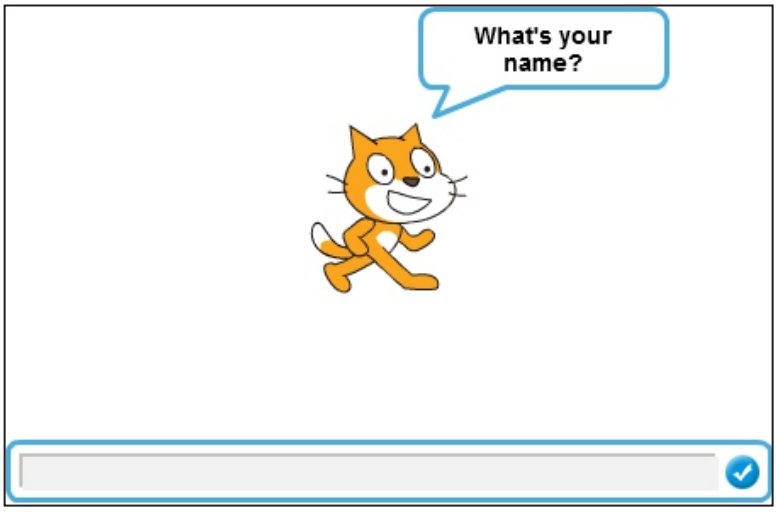
Chapter 6



Advancing with Scratchy

After completing this chapter you will be able to know how to use the following blocks below:

- **Control blocks;**
- **Sensing blocks and Operators blocks.**

Advanced Concepts

Now that you've mastered the basics of the Scratch programming language, you're ready to tackle some more advanced concepts. In this chapter, you will investigate how to work with user input, with control blocks that enable your project to take different actions depending on circumstances, with Boolean blocks that evaluate whether conditions are true or false, and with operator blocks that manipulate values. Along the way, you'll learn some important concepts that are common to all high-level programming languages, but are still as simple to use as dragging and snapping blocks in Scratch. Your first step is to take a closer look at the blocks in the **Control, Sensing**, and **Operators** categories in the block palette.

Control Blocks

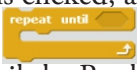
Blocks in the **Control** category (see Figure 6-1) literally control the flow of the action within your scripts by looping repetitively through a sequence of instructions, or by evaluating and reacting to conditions, or by pausing or stopping the action entirely.

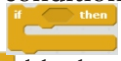
You're already very familiar with three blocks housed here: the 

block, which repeats the sequence of actions within it a specified number of times to create what is called a loop in programming language; the 

block, which repeats (loops) the sequence of actions forever, or until the script is stopped manually; and the 

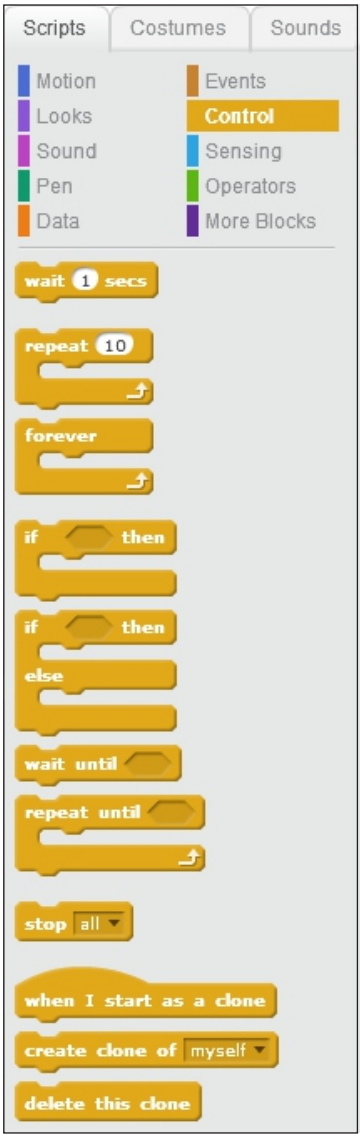
block, which pauses the script for the specified time. These blocks, however, are only the beginning of the programming power and flexibility that the Control category offers. When combined with Boolean blocks, several more Control blocks add flexibility to your scripts, enabling you to issue instructions based on specific conditions. As you remember from Chapter 2, Boolean blocks evaluate conditions and then reports them as either true or false. Because you can snap Boolean blocks inside Control blocks, you can then tell your script to take an action when a certain condition is met. For example, the 

block pauses the script until the Boolean block that you snap into it reports its condition as true. So, instead of waiting for a specific number of seconds, you can use this block to wait until the mouse is clicked, a value is entered, or a timer reaches a certain value. Similarly, the 

block continues repeating the sequence of actions within it until the Boolean block that you snap into it reports its condition as true. This type of loop is called a while loop in traditional programming language, because it repeats while a specific condition exists. Sometimes you might want to perform a different action depending on the circumstances in your script. For instance, if a user answers "yes" to a question, then you would want to take a different action than if the answer is "no," or you might want the pen to draw only when the user holds down the mouse button. To react to circumstances this way, you need what is called a conditional statement or an If/Then statement in programming language and the 

block in Scratch. If the Boolean block that you snap into the 

block evaluates as true, then this C block executes the instructions (or sequence of instructions) within it. If the condition is false, the script skips to the block immediately after the C block and executes it. The block works in a similar way. If the Boolean block that you snap into it reports its condition as true, then the sequence of instructions in the first part of the block executes. If the



Script 6-1. Control blocks

condition is false, the script ignores the first sequence of blocks and executes the second sequence of instructions within the block after the word else. The last block in the Control category, the stop all block, stops scripts in a project from executing any instructions. You can use the pull down menu to select all scripts in a project (as in the example) or just one particular script.

Sensing Blocks

The **Sensing** category (see Figure 6-2) contains a combination of Boolean blocks and reporter blocks that help you sense and evaluate what the user of your script and your sprites are doing. Many of these blocks snap into, or embed, in the various **Control** or **Operators** blocks, enabling your script to take action depending on their value or condition.

The four Boolean blocks all evaluate a condition that you specify, and then return a value of either true or false to the block they're embedded into. For example, the touching ? block evaluates whether the sprite touches the item that you choose from the pull-down menu: the mouse-pointer, the edge of the stage, or another sprite. The touching color ? block returns a value of true if the sprite is touching the specified color. You select the color by clicking in the color box in the block and then clicking the desired color anywhere in the Scratch interface. The key space pressed? block checks if the specified key is pressed. You can select which key to test for, by using the pull down menu. The last Boolean block, mouse down?, checks if the primary mouse button is held down. The Sensing category also contains six reporter blocks that store information about the mouse position, sprite position, user input, and more. You can then embed these blocks into to other blocks to pass on that information for evaluation and action. For instance, the mouse x and mouse y blocks, respectively, hold the mouse-pointer's current X and Y coordinates on the stage. Using its two pull-down menus, you can set the x position of Sprite1 block to hold the value of a specified sprite's X or Y coordinate, direction, or custom name, among other choices. This timer block holds the timer value. The timer starts running from 0.0 as soon as Scratch is launched and increases by 1 every second; the reset timer block sets the timer value back to 0.0. The username block holds the username of the user who is logged on to Scratch. Finally, a pair of blocks enable you to ask the user a question, and then store the answer. The ask What's your name? and wait block asks the question that you type in its text field, opens an input window on the stage, and waits for the user's input. Scratch then saves the user's input in the answer block.

Figure 6-2. Sensing blocks

Operators Blocks

The blocks in the **Operators** category (see Figure 6-3) do just what their name implies: Perform operations on the values supplied to them. Sometimes the operations are mathematical, then the block reports the resulting value. Other blocks are Boolean blocks that report whether the indicated operation produced a true or false result. **Operators** blocks all embed into others blocks, which then take action based on the values they report. Several blocks in this category perform math then report the result. For instance, the + block adds the two values that you supply, the - block subtracts two values, and the / block divides two values. The mod block reports the remainder when the first value is divided by the second specified value. For example, when 7 is divided by 3, a remainder of 1 is reported. The round block rounds the specified number to the nearest integer.

Decimals that are 0.5 or higher are rounded up. Decimals that 0.5 or less are rounded down, so if you enter 5.3, this block reports 5. The sqrt of 9 block reports the result of the selected operation on the specified value. The default is the square root operation, but clicking the pull-down menu displays other mathematical operation choices. In the previous activity, the block will report the square root of 9, which is 3. Finally, the pick random 1 to 10 block picks a random value between the two specified numbers and reports the result. A second group of blocks in the Operators category perform Boolean comparisons and return a value of true or false. For instance, < determines whether the first value is smaller than the second value. If it is, the block reports true, otherwise it reports false. The > block does the opposite, determining whether the first value is larger than the second value. The = block reports true if the two values are equal. Two blocks, and and or, enable you to combine and evaluate two Boolean blocks, then return true or false. Both embedded blocks on each side of the and block need to return true for the condition represented by this block to be true. If one returns false, then this block returns false. For example, on the left side, you use 3 < 4, and on the right side, you use 1 < 3.Both of these Boolean blocks evaluate to true because 3 is less than 4 and 1 is less than 3. Because both sides are true, the and block evaluates to true. Only one side of the or block needs to be true, however, for this block to report true. For example, on the left side you use 3 < 4, and on the right side, you use 5 < 3.

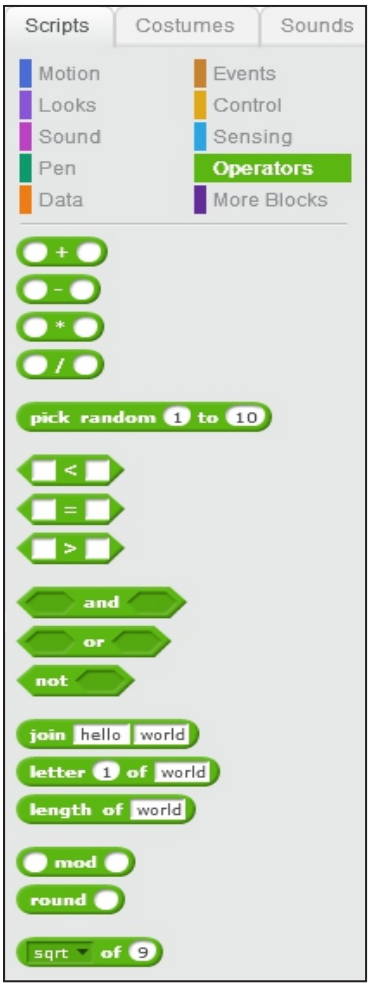


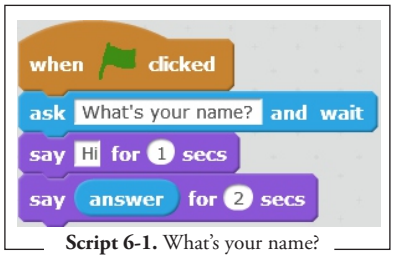
Figure 6-3. Operators blocks

The left Boolean block evaluates to true because 3 is less than 4, but the right Boolean block evaluates to false, because 5 is less than 3. Because only one side needs to be true, the or still evaluates to true. The not block returns the opposite of the result returned by the block embedded within it. If the block embedded within the not block reports true, therefore, the not block returns false and vice versa. The final three blocks in the Operators category work with text. The join hello world block joins the two values that you specify in the block and reports the result. This example returns hello world. The letter 1 of world block reports the character in the specified position in the supplied text. The value can consist of text, number, symbol, or even a space. In the previous activity, it reports w, which is the first letter of the supplied word, world. The length of world block reports how many characters the specified value contains.

Activities

Keeping track of three new categories of blocks is a challenge, but trying some examples that put them to use will help. These activity scripts are more dynamic than the ones from previous chapters. You'll practice making scripts that interact with the user, check a password, draw with the mouse-pointer, and produce different results depending on the condition that's evaluated.

Activity 6-1: What's Your Name?



Script 6-1. What's your name?

This activity uses Sensing blocks to gather and report on information the user enters. Specifically, the script first asks the user to enter his or her name, then the script displays that name in a speech bubble. When asked for user input (see Figure 6-4), enter a name and press the Enter key on your keyboard. Script 6-1 starts running when the user clicks the green flag. The next block creates a speech bubble that displays What's your name?, opens a user input field, and waits for the user's input (see Figure 6-4). The next block creates a speech bubble that displays Hi for 1 second. To create the last block, snap the say [ ] for 2 secs block onto the script, then drag the answer block over its entry field to embed the Sensing block into the Sound block. Now, the embedded block reports the reader's answer to the say [ ] for 2 secs block, which creates a speech bubble for the sprite and displays the user's input for 2 seconds. **Table 6-1** lists the blocks and describes the actions used in this activity.

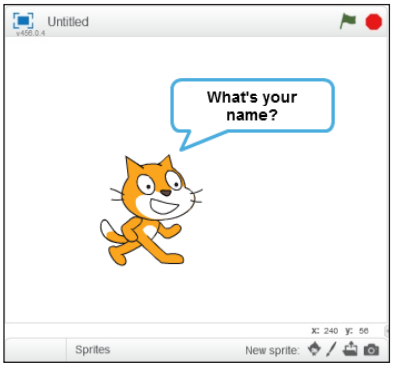


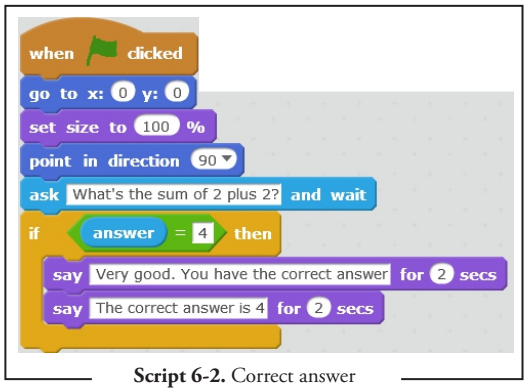
Figure 6-4. User input

Table 6-1. Code Blocks in What's Your Name?

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
ask What's your name? and wait	The sprite gets a speech bubble that displays What's your name?, opens a user input field, and waits for user input.
say Hi for 1 secs	The sprite gets a speech bubble that displays Hi for 1 second.
say [ ] for 2 secs	The sprite gets a speech bubble that displays the specified text for 2 seconds.
answer	Holds the current user input value.

Activity 6-2: What's The Correct Answer?

When you combine a C block conditional statement, a sensing block, and an operator, you can produce sophisticated actions with ease. To demonstrate, this script asks the user an easy math question (see Figure 6-5), and then stores and evaluates the answer. If the user answers correctly, then script will display some text in a speech bubble. If the answer is wrong, the script stops running. Although this particular math question is simple, the underlying technique is one that you can use often in your scripts.



Script 6-2. Correct answer

Script 6-2 starts running when the user clicks the green flag. The next three blocks move the sprite to the center of the stage, set the sprite to its default size, and make the sprite face to the right. The next block creates a speech bubble that displays What's the sum of 2 plus 2?, opens a user input field, and waits for the user's input. The C block is a conditional statement that checks if the user's input is equal to 4. If the condition is true, the script will execute the sequence

of actions within the C block. If the condition is false, the script stops running. To create this conditional statement, first drag the blocks to the scripts area and embed the user input block answer in the left field of the = 4 block, and then type 4 in its right field. Next, drag the operator block into if then .Block in the scripts area. The first block within the conditional statement creates a speech bubble that displays Very good. You have the correct answer for 2 seconds. The last block also creates a speech bubble for the sprite and displays The correct answer is 4 for 2 seconds. **Table 7-2** lists the blocks and describes the actions used in this activity.

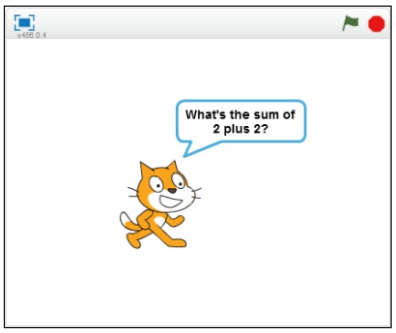


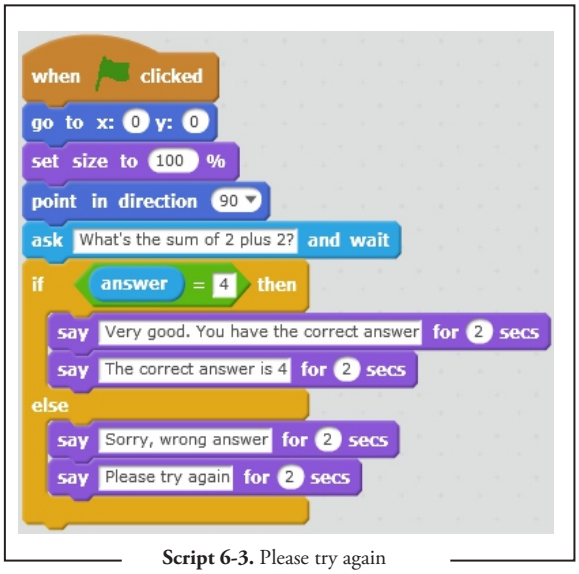
Figure 6-5. Enter correct answer

Table 6-3. Code Blocks in Please Try Again

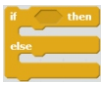
Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
go to x: 0 y: 0	Move the sprite to the position where X = 0 and Y = 0, which is the center of the stage.
set size to 100 %	Set the sprite to its default size.
point in direction 90	Make the sprite face to the right.
ask What's the sum of 2 plus 2? and wait	The sprite gets a speech bubble that displays What's the sum of 2 plus 2?, opens a user input field and waits for user input.
if then	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.
= 4	If the value on the left side is equal to 4, then this condition is true; otherwise, the condition is false.
answer	Hold and report the current user input value.
say Very good. You have the correct answer for 2 secs	The sprite gets a speech bubble that displays Very good. You have the correct answer for 2 seconds.
say The correct answer is 4 for 2 secs	The sprite gets a speech bubble that displays The correct answer is 4 for 2 seconds.

Activity 6-3: Please Try Again

In activity 6-2, you created an If/Then conditional statement. This time, try the  's If/Then/Else conditional statement to control the action in your script based on user input. The script asks the user an easy math question, and then displays a different set of speech bubbles, depending on whether the user answers correctly or incorrectly.

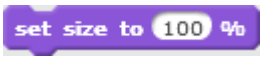
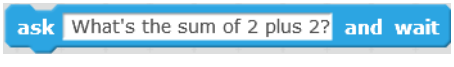
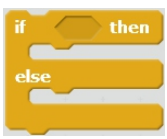


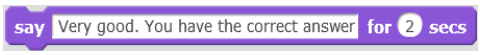
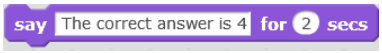
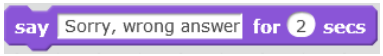
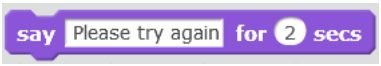
Script 6-3. Please try again

Script 6-3 starts running when the user clicks the green flag. The next three blocks move the sprite to the center of the stage, set the sprite to its default size, and makes the sprite face to the right. The next block creates a speech bubble that displays What's the sum of 2 plus 2?, opens a user input field, and waits for the user's input. The next block is a conditional statement that checks if the user's input is equal to 4. If the condition is true, the  script will execute the first sequence of actions within the Then sequence). The first block in the Then sequence creates a speech bubble that displays Very good. You have the correct answer for 2 seconds. The second block also creates a speech bubble for the sprite and displays The correct answer is 4 for 2 seconds. If the condition is false, the script executes the second sequence of blocks contained after the word else. The first block in the Else sequence creates a speech bubble that displays Sorry, wrong answer for 2 seconds. The second block in the sequence creates a speech bubble that displays Please try again for 2 seconds. **Table 6-3** lists the blocks and describes the actions used in this activity.

OUTPUT

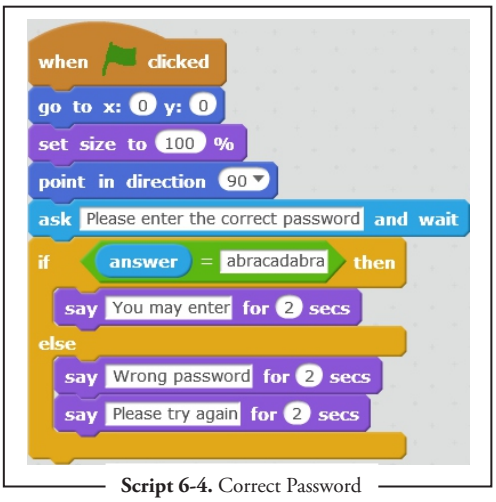
Table 6-3. Code Blocks in Please Try Again

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Move the sprite to the position where X = 0 and Y = 0, which is the center of the stage.
	Set the sprite to its default size.
	Make the sprite face to the right.
	The sprite gets a speech bubble that displays What's the sum of 2 plus 2?, opens a user input field and waits for user input.
	Check if the condition is true. If the condition is true, execute the actions within it, before the word else. If the condition is false, execute the actions after the word else within the block.

	If the value on the left side is equal to 4, then this condition is true; otherwise, the condition is false.
	Hold and report the current user input value.
	The sprite gets a speech bubble that displays Very good. You have the correct answer for 2 seconds.
	The sprite gets a speech bubble that displays The correct answer is 4 for 2 seconds.
	The sprite gets a speech bubble that displays Sorry, wrong answer for 2 seconds.
	The sprite gets a speech bubble that displays Please try again for 2 seconds.

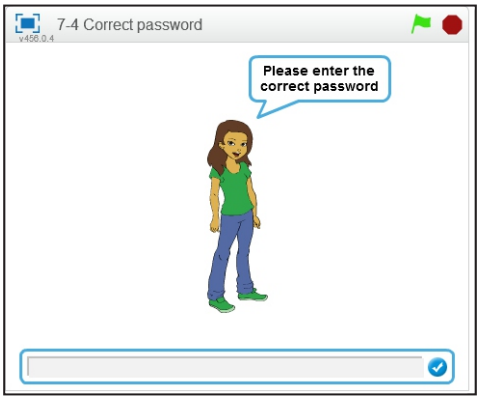
Activity 6-4: Enter Correct Password

Script 6-4 uses the same technique as you used in the previous activity, but this time it evaluates a text answer from the user instead of a number. The script asks the user to enter the correct password. If the user enters abracadabra, the script will display some text in a speech bubble, announcing that the answer is right and You may enter. If the answer is wrong, the script will also display some text announcing that the answer is wrong and to please try again.



Script 6-4. Correct Password

OUTPUT

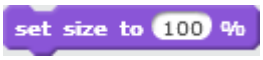


The first four blocks of Script 6-4 are similar to the previous script. The script starts running when the user clicks the green flag. The next three blocks move the sprite to the center of the stage, set the sprite to its default size, and make the sprite face to the right. The next block creates a speech bubble that displays Please enter the correct password. It also opens a user input field and waits for the user's input. The next block is an If/Then/Else conditional statement that checks if the user's input is equal to the word abracadabra. If the condition is true, the script will execute the Then sequence (the next block within the C block). In this case, it is one block that creates a speech bubble that displays You may enter for 2 seconds. If the condition is false, the script executes the two blocks after the word else. The first block of the pair creates a speech bubble that displays Wrong password for 2 seconds. The second creates a speech bubble that displays Please try again for 2 seconds. **Table 6-4** lists the blocks and describes the actions used in this activity.

Table 6-4 Code Blocks in Enter the Correct Password

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.

Move the sprite to the position where X = 0 and Y = 0, which is the center of the stage.



Set the sprite to its default size.

Make the sprite face to the right.

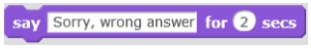
The sprite gets a speech bubble that displays Please enter the correct password, opens a user input field, and waits for user input.

Check if the condition is true. If the condition is true, execute the actions within it, before the word else. If the condition is false, execute the actions after the word else within the block.

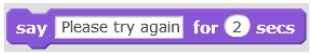
If the value on the left side is equal to abracadabra, then this condition is true; otherwise, the condition is false.

Hold and report the current user input value.

The sprite gets a speech bubble that says You may enter for 2 seconds.



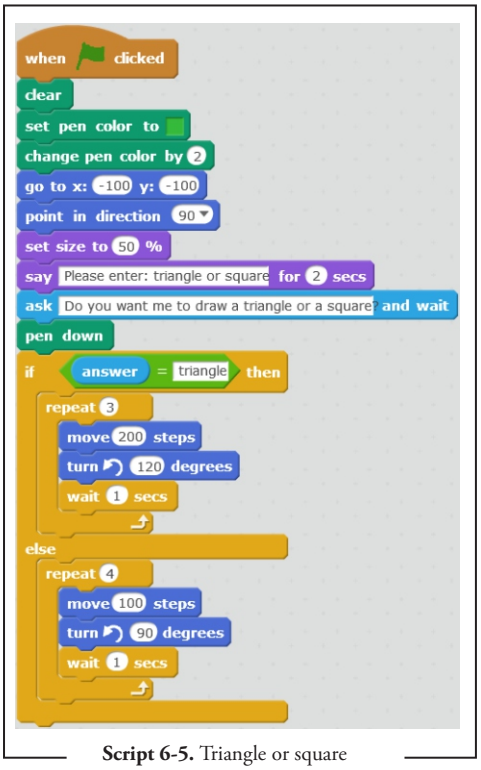
The sprite gets a speech bubble that displays Sorry, wrong password for 2 seconds.



The sprite gets a speech bubble that displays Please try again for 2 seconds.

Activity 6-5: Triangle or Square

Remember, trying to figure out what the script does before you create and run it is a good way to learn and understand the blocks and techniques the script uses. Study Script 6-5 for a moment. What do you think it does? Do any of the techniques look familiar from past chapters? They should! Script 6-5 asks the user whether to draw a triangle or a square. Based on the user's answer, either a triangle or a square is drawn using the repeating C block technique that you learned in Chapter 3. Script 6-5 starts running when the user clicks the green flag. The second block of code clears the stage of any marks made by the pen or stamp. The third block sets the pen color, and the next block changes the pen size. The next block moves the sprite to the coordinates, (–100, –100). The next block makes the sprite face to the right. The next block sets the sprite to half of its default size, and the block after that creates a speech bubble for the sprite and displays the text Please enter triangle or square for 2 seconds.



Script 6-5. Triangle or square

The next block creates a speech bubble with the text Do you want me to draw a triangle or a square?, opens a user input field, and waits for the user's input.

The pen down block means that the pen is on the stage and is ready to draw. The next block is an If/Then/Else conditional statement. If the user input stored in the answer block is equal to triangle, then the condition is true the script executes the Then sequence of actions (in the top half of the block. The sequence draws a triangle using a C block that repeats the sequence of actions within it 3 times: Move the sprite 200 pixels in the direction that it's facing to draw with the pen for 200 pixels, rotates the sprite 120 degrees counterclockwise, and then pauses for 1 second. If the answer block is not equal to triangle, the condition is false and the Else sequence of actions are executed. The first one is a repeat 4 block that repeats the sequence of actions within it 4 times: Move the sprite 100 pixels in the direction that it's facing, rotate the sprite 90 degrees counterclockwise, and then pause for 1 second. The last block of this script is pen up, which lifts the pen off the stage so if the sprite moves, the pen will not draw.

OUTPUT

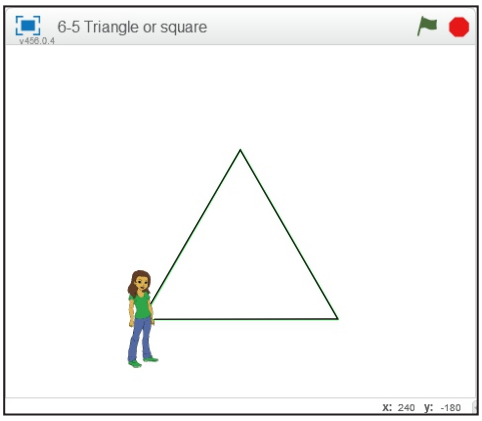


Table 6-5. Code Blocks in Triangle or Square

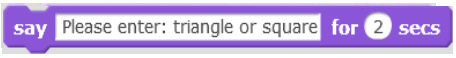
Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
clear	Remove all marks previously made by the pen or stamp.
set pen color to	Set the pen color to lime color.

Change the pen size to 2.

Move the sprite to coordinates (X = –100, Y = –100).

Make the sprite face to the right.

Set the sprite to half its default size.



The sprite gets a speech bubble that displays Please enter: triangle or square for 2 seconds.

The pen is on the stage and is ready to draw.

Check if the condition is true. If the condition is true, execute the actions within it, before the word else. If the condition is false, execute the actions after the word else within the block.

If the value on the left side is equal to triangle, then this condition is true; otherwise, the condition is false.

Hold and report the current user input value.

Repeat the actions represented by the blocks within this block three times.

Move the sprite 200 pixels in the direction that it's facing.

Turn the sprite counterclockwise 120 degrees.

The script waits 1 second. No actions are performed for 1 second.

Repeat the actions represented by the blocks within this block four times.

Move the sprite 100 pixels in the direction that it's facing.

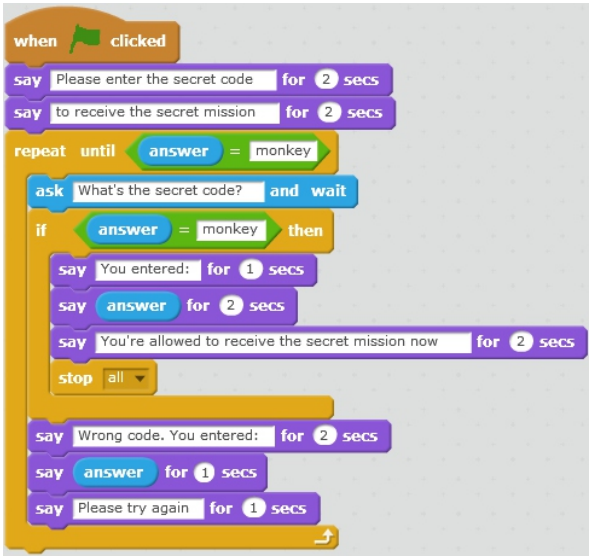
Turn the sprite counterclockwise 90 degrees.

The script waits 1 second. No actions are performed for 1 second.

The pen is off the stage and cannot draw.

Activity 6-6: Secret Mission

This activity puts the repeat until and stop all blocks to work in a script that prompts the user to enter a password, tests it, then gives the user as many attempts as needed to enter the correct password.



Script 6-6. Secret mission

Script 6-6 also starts running when the user clicks the green flag. The next two blocks both create two-second speech bubbles. First Please enter the secret code is displayed and then to receive the secret mission. Next, the C block repeats the sequence of actions within it until its condition is true, meaning the user input stored in answer is equal to monkey. The first block in the sequence creates a speech bubble containing What's the secret code?, opens a user input field, and waits for the user's input. The next block is an If/Then conditional statement that also checks whether answer is equal to monkey. If the condition is true, then the first block within the If/Then conditional statement C block will display a speech bubble with You entered: for 1 second. The next block also creates a speech bubble that displays the user input for 2 seconds. The next block creates a third speech bubble with You're allowed to receive the secret mission now for 2 seconds. The last block within this sequence stops all scripts in this project from running. If answer is not equal to monkey, then the final three blocks in the execute, displaying three consecutive speech bubble that contain Wrong code, You entered, the current user input, and Please try again, and then the loop repeats. The technique shown in this script can be used any time that you need to test user input for a specific answer. **Table 6-6** lists the blocks and describes the actions used in this activity.

OUTPUT

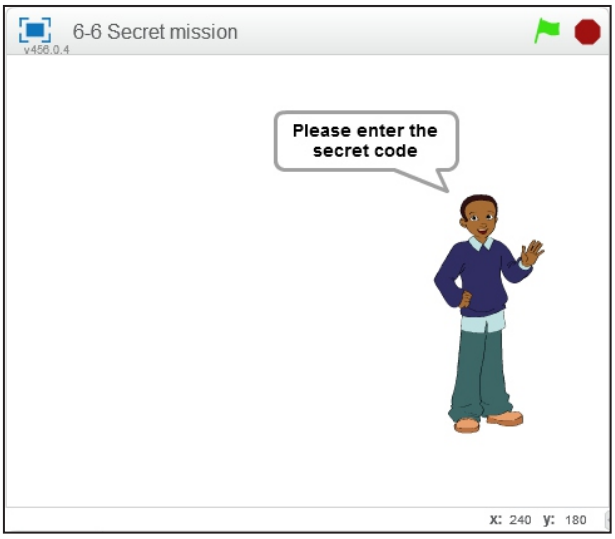


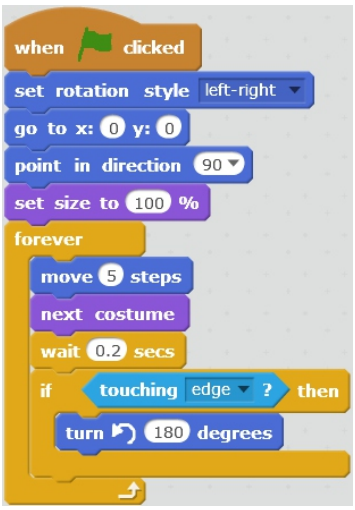
Table 6-6. Code Blocks in Secret Mission

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
say Please enter the secret code for 2 secs	The sprite gets a speech bubble that displays Please enter the secret code for 2 seconds.
say to receive the secret mission for 2 secs	The sprite gets a speech bubble that displays to receive the secret mission for 2 seconds.
repeat until	Execute the blocks/instructions within this block until the specified condition is true.
= monkey	If the value on the left side is equal to monkey, then this condition is true; otherwise, the condition is false.

answer	Hold and report the current user input value.
ask What's the secret code? and wait	The sprite gets a speech bubble that displays What's the secret code?, opens a user input field, and waits for user input.
if then	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.
= monkey	If the value on the left side is equal to monkey, then this condition is true; otherwise, the condition is false.
answer	Hold and report the current user input value.
say You entered: for 1 secs	The sprite gets a speech bubble that displays You entered: for 1 second.
say for 2 secs	The sprite gets a speech bubble that displays the specified text for 2 seconds.
answer	Hold and report the current user input value.
say You're allowed recive the secret mission new for 2 secs	The sprite gets a speech bubble that displays You're allowed to receive the secret mission now for 2 seconds.
stop all	Stop all scripts in this project.
say Wrong code.You entered: for 2 secs	The sprite gets a speech bubble that displays Wrong code. You entered: for 2 seconds.
say for 1 secs	The sprite gets a speech bubble that displays the specified text for 1 second.
answer	Stop all scripts in this project.
say Please try again for 1 secs	The sprite gets a speech bubble that displays Please try again for 1 second.

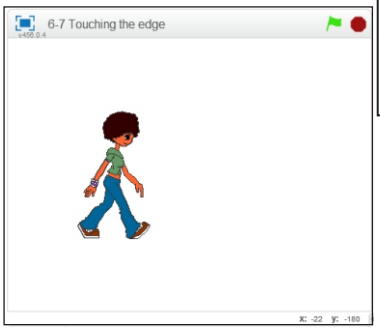
Activity 6-7: Touching the Edge?

You've used the if on edge, bounce block before, but in this activity, you're going to script a version of it yourself using your new conditional statement skills. Script 6-7 starts running when the user clicks the green flag. The next four blocks set the rotation style of the sprite to **left-right**, move the sprite to the center of the stage, make the sprite face to the right, and set the sprite to its default size. Next is a C block that executes the sequence of actions within it forever. The first three blocks in the sequence move the sprite 5 pixels in the direction that it's facing, change the sprite's costume, and pause the script for 0.2 seconds. Next, there is an If/Then conditional statement that evaluates whether the sprite is touching the edge of the stage. If it is (a true result), then the sprite rotates 180 degrees counterclockwise. If the sprite isn't touching an edge (a false result), the action loops back to the move 5 steps block and the sequence repeats.



Script 6-7. Touching the edge

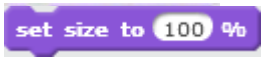
OUTPUT



The movement and costume changes give the illusion as if the sprite is walking. If the sprite hits the edge of the stage, the If/Then conditional statement turns the sprite and moves it in the opposite direction, just like the 

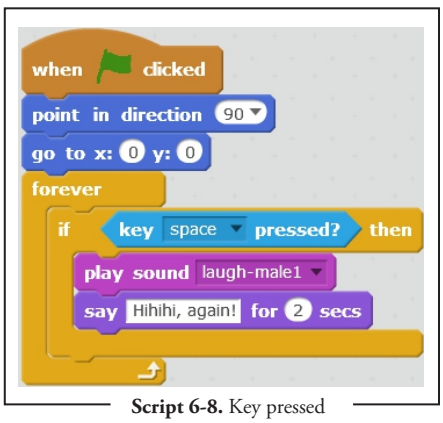
block. **Table 6-7** lists the blocks and describes the actions used in this activity.

Table 6-7. Code Blocks in Touching the Edge?

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Set the rotation style of the sprite so that it can only face left or right.
	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
	Make the sprite face to the right.
	Set the sprite to its default size.
	Repeat all the instructions/blocks within this block forever.
	Move the sprite 5 pixels in the direction that it's facing.
	Change the sprite to the next costume in the Costumes tab.
	The script waits 0.2 seconds. No actions are performed for 0.2 seconds.
	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.
	Checks if the sprite is touching the edge of the stage. If it is, then this condition is set to true.
	Turn the sprite counterclockwise 180 degrees.

Activity 6-8: Key Pressed?

Have you already figured out what Script 6-8 does? This activity checks whether the space bar is pressed. Every time the space bar is pressed, the script plays a sound and creates a speech bubble with text. Try it out.



Script 6-8. Key pressed

The script starts running when the user clicks the green flag. The next two blocks make the sprite face to the right and move the sprite to the center of the stage. The next block is a C block that loops through the sequence of actions within it forever. That sequence is an If/Then conditional statement that first evaluates whether the space bar is pressed.

If it is (a true result), then the **laugh-male1** sound plays and a Hihihi, again! speech bubble displays for 2 seconds.

If the space bar is not pressed, the action loops back to check for another key press. **Table 6-8** lists the blocks and describes the actions used in this activity.

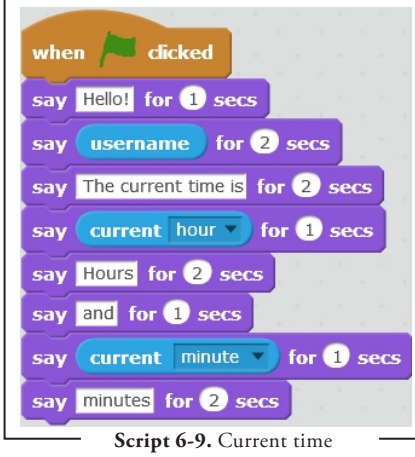
OUTPUT

Table 6-8. Code Blocks in Key Pressed?

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
point in direction 90	Make the sprite face to the right.
go to x: 0 y: 0	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
forever	Repeat all the instructions/blocks within this block forever.
if then	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.
key space pressed?	Check if the space bar is clicked. If it is, then this condition is set to true.
play sound laugh-male1	Play the sound **laugh-male1.**
say Hihihi, again! for 2 secs	The sprite gets a speech bubble that displays Hihihi, again! for 2 seconds.

Activity 6-9: Current Time

This activity creates some speech bubbles that display the user's name, the current hour, and the current minute, along with other text, to demonstrate two of the reporter blocks in the Sensing category.



Script 6-9. Current time

Script 6-9 starts running when the user clicks the green flag. The next block creates a speech bubble that displays Hello! for 1 second. The next block creates a speech bubble and displays the username of the user who is logged into Scratch for 2 seconds.

The next pair of blocks create a speech bubble that displays the current time is followed by a speech bubble that displays the current hour. The next pair of blocks creates speech bubbles that display hours and then and the last two blocks create speech bubbles that display the current minute and minutes for 2 seconds. **Table 6-9** lists the blocks and describes the actions used in this activity.

OUTPUT

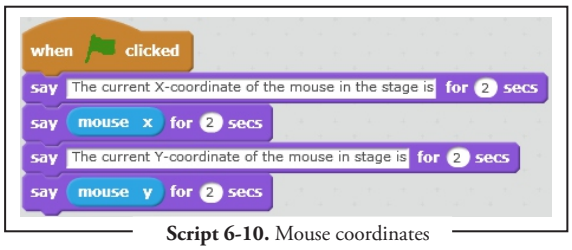
Table 6-9. Code Blocks in Current Time

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
say Hello! for 1 secs	The sprite gets a speech bubble that displays Hello! for 1 seconds.
say [ ] for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
username	Hold and report the username of the user who has logged in Scratch.
say The current time is for 2 secs	The sprite gets a speech bubble that displays The current time is for 2 seconds.
say [ ] for 1 secs	The sprite gets a speech bubble that displays the specified text or value for 1 second.
current hour	Hold and report the current hour.
say hours for 2 secs	The sprite gets a speech bubble that displays hours for 2 seconds.
say and for 1 secs	The sprite gets a speech bubble that displays and for 1 seconds.
say [ ] for 1 secs	The sprite gets a speech bubble that displays the specified text or value for 1 second.
current minute	Hold and report the current minute.
say minutes for 2 secs	The sprite gets a speech bubble that displays minutes for 2 seconds.

Activity 6-10: Mouse Coordinates

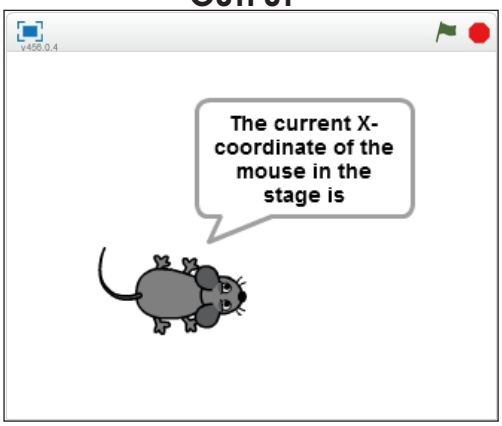
This activity demonstrates how to provide the current X and Y coordinates of the mouse to the code blocks, and then display them on the screen in the stage area.

Script 6-10 also starts running when the user clicks the green flag. The second block creates a speech bubble that displays the current X-coordinate of the mouse in the stage is for 2 seconds. The third block also creates a speech bubble and displays the current X coordinate of the mouse on the stage. The last two blocks do the same for the Y coordinate of the mouse on the stage.



Script 6-10. Mouse coordinates

OUTPUT



You can use this technique whenever you need to report or evaluate the X and Y coordinates. For example, you can use mouse x mouse y or both blocks in a Boolean block to create a condition and make a decision whether the condition is true or false. For example, if you need to test whether the mouse is in a certain area of the stage, you could evaluate if the X coordinate is greater than 50; for example, by using the mouse x block in a Boolean block. **Table 6-10** lists the blocks and describes the actions used in this activity.

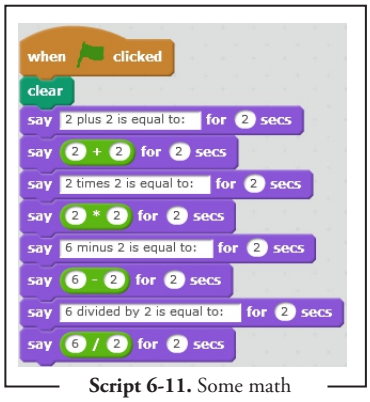
Table 6-10 Code Blocks in Mouse Coordinates

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
say The current X-coordinates of the mouse in the stage is for 2 secs	The sprite gets a speech bubble that displays The current X-coordinate of the mouse in the stage is for 2 seconds.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
mouse x	Hold and report the mouse-pointer current X coordinate on the stage.
say The current Y-coordinate of the mouse in stage is for 2 secs	The sprite gets a speech bubble that displays The current Y-coordinate of the mouse in stage is for 2 seconds.
say for 2 secs	The sprite gets a speech bubble that displays The current Y-coordinate of the mouse in stage is for 2 seconds.
mouse y	Hold and report the mouse-pointer current Y coordinate on the stage.

Activity 6-11: Let's Do Some Math

This activity demonstrates several mathematical blocks from the Operators category, providing the answers to their equations as input for say for 2 secs blocks.

Script 6-11 starts running when the user clicks the green flag. The second block of code clears the stage, removing any marks made by the pen or stamp. This block is not necessary, but it makes sure you start with a clean stage. The next pair of block create a speech bubble that displays 2 plus 2 is equal to: for 2 seconds, then create a speech bubble that displays the result of the (2 + 2) operator block. Remember, you need to drag and embed the (2 + 2) into the text field of the say for 2 secs block to provide the result of one as the input to the other. The next three pairs of blocks create a speech bubble that introduce and display the answers for simple multiplication, subtraction, and division questions using the respective operator blocks: (2 * 2) (6 - 2) and (6 / 2). **Table 6-11** lists the blocks and describes the actions used in this activity.



Script 6-11. Some math

OUTPUT

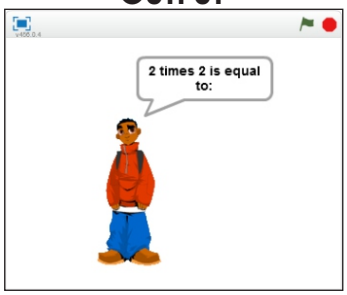


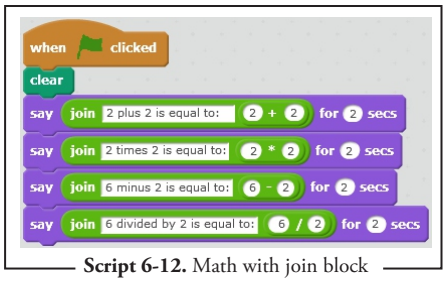
Table 6-11 Code Blocks in Let's Do Some Math

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
clear	Remove all marks previously made by the pen or stamp.
say 2 plus 2 is equal to: for 2 secs	The sprite gets a speech bubble that displays 2 plus 2 is equal to: for 2 seconds.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
2 + 2	Add two values (2 + 2) and report the result.
say 2 times 2 is equal to: for 2 secs	The sprite gets a speech bubble that displays 2 times 2 is equal to: for 2 seconds.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
2 * 2	Multiply two values (2 * 2) and report the result.
say 2 times 2 is equal to: for 2 secs	The sprite gets a speech bubble that displays 6 minus 2 is equal to: for 2 seconds.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
6 - 2	Subtract one value from another (6 – 2) and report the result.
say 6 minus 2 is equal to: for 2 secs	The sprite gets a speech bubble that displays 6 divided by 2 is equal to: for 2 seconds.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
6 / 2	Divide one value by another (6 / 2) and report the result.

Activity 6-12: Math with the Join Block

A more efficient way to produce the same results as Script 6-11 is to use the join 2 plus 2 is equal to: you can join two values, and then embed them together into another block. Script 6-12, for example, joins the descriptive text about an equation with its result and embeds both pieces of information into a single say for 2 secs block. The script is now half the size of the previous version!

Script 6-12 starts running when the user clicks the green flag. The second block of code clears the stage area of any marks made by the pen or stamp. Next is the first three-block combination: the 2 + 2 operator block is embedded in the join 2 plus 2 is equal to: block. This newly joined pair is then embedded in the say for 2 secs block in the scripts area. Together they create a speech bubble that displays the text 2 plus 2 is equal to: and 4 (the sum of 2 + 2) for 2 seconds. The next block is also a three-in-one block that creates a speech bubble that displays the text 2 times 2 is equal to: and the result of the operator block for 2 seconds. The technique is repeated for the next two blocks, joining the appropriate text with the 6 - 2 and 6 / 2 operators. **Table 6-12** lists the blocks and describes the actions used in this activity.



Script 6-12. Math with join block

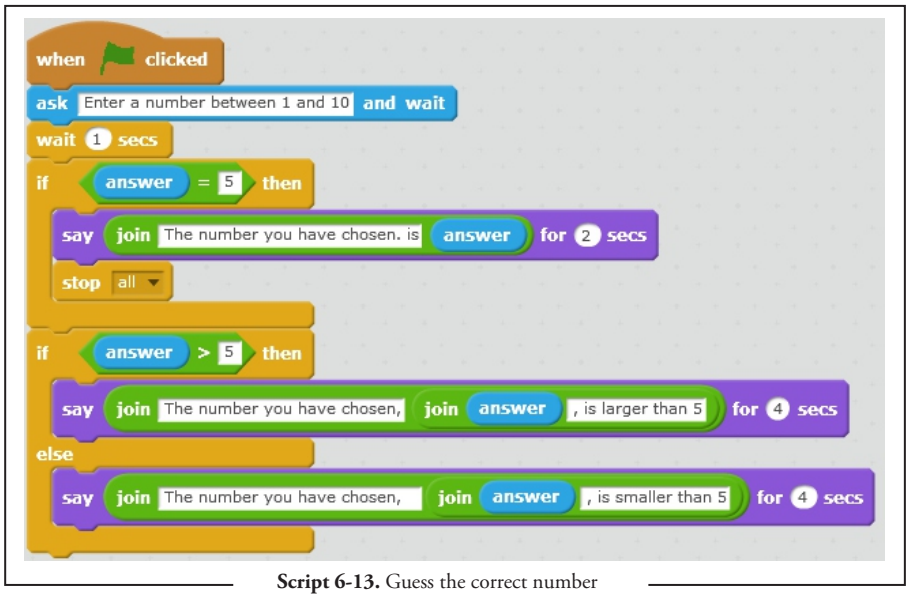
OUTPUT

Table 6-12. Code Blocks in Math with the Join Block

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
clear	Remove all marks previously made by the pen or stamp.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
join 2 plus 2 is equal to:	Join the two values that have been specified in the block and report the result.
2 + 2	Add two values (2 + 2) and report the result.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
join 2 times 2 is equal to:	Join the two values that have been specified in the block and report the result.
2 * 2	Multiply two values (2 * 2) and report the result.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
join 6 minus 2 is equal to:	Join the two values that have been specified in the block and report the result.
6 - 2	Subtract one value from another (6-2) and report the result.
say for 2 secs	The sprite gets a speech bubble that displays the specified text or value for 2 seconds.
join 6 divided by 2 is equal to:	Join the two values that have been specified in the block and report the result.
6 / 2	Divide one value by another (6 / 2) and report the result.

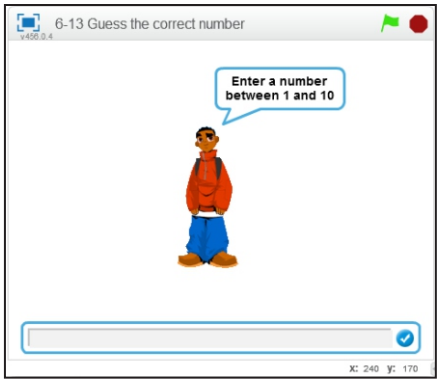
Activity 6-13: Guess the Correct Number

This script demonstrates more ways of joining blocks to produce shorter, more efficient scripts. This activity script creates a guessing game. The script asks the user to enter a number between 1 and 10. The correct number to be guessed is 5. The script displays different text, depending on the number that the user guessed—whether it's the correct number, larger than 5, or smaller than 5.



Script 6-13. Guess the correct number

OUTPUT



Script 6-13 starts running when the user clicks the green flag. The next block creates a speech bubble, displays Enter a number between 1 and 10, opens a user input field, and waits for the user's input. The next block pauses the script for 1 second. The  block checks if the user input equals 5. If it does (a true result), the actions within the C block executed: a speech bubble displays the text The number you have chosen, is and the user input for 2 seconds, then the  block stops all scripts in this project from running. If the user input is not equal to 5 (a result of false), the script skips to the If/Then/Else conditional statement block to check if  is greater than 5. If it is, the script executes the actions in the Then portion of the block. Here a four-in-one combination of blocks creates a speech bubble that displays The number you have chosen, the user input, and is larger than 5 for 4 seconds. If  is not greater than 5, the actions in the Else portion of the block are executed instead: A speech bubble displays The number you have chosen, the user input, and is smaller than 5 for 4 seconds.

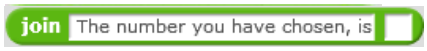
Table 6-13. Code Blocks in Guess the Correct Number

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	The sprite gets a speech bubble that displays Enter a number between 1 and 10, opens user input field and waits for user input.
	The script waits 1 second. No actions are performed for 1 second.
	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.

If the value on the left side is equal to 5, then this condition is true; otherwise, the condition is false.

Hold and report the current user input value.

The sprite gets a speech bubble that displays the specified text or value for 2 seconds.



Join the two values that have been specified in the block and report the result.

Hold and report the current user input value.

Stop all scripts in this project.

Check if the condition is true. If the condition is true, execute the actions within it, before the word else. If the condition is false, execute the actions after the word else within the block.

If the value on the left side is greater than 5, then this condition is true; otherwise, the condition is false.

Hold and report the current user input value.

The sprite gets a speech bubble that displays the specified text or value for 4 seconds.

Join the two values that have been specified in the block and report the result.

Join the two values that have been specified in the block and report the result.

Hold and report the current user input value.

The sprite gets a speech bubble that displays the specified text or value for 4 seconds.

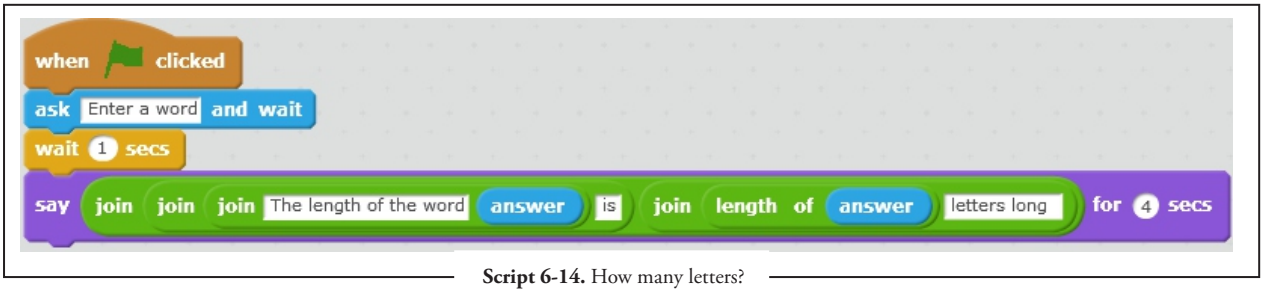
Join the two values that have been specified in the block and report the result.

Join the two values that have been specified in the block and report the result.

Hold and report the current user input value.

Activity 6-14: How Many Letters in the Word?

Sometimes you may need to determine the number of characters in a word entered by the user, such as for a password that must be at least of a specific length. This activity uses the length of and answer blocks to do just that.



Script 6-14. How many letters?

Script 6-14 starts running when the user clicks the green flag. The next block creates a speech bubble, displays the text Enter a word, opens a user input field, and waits for the user's input. The next block pauses the script for 1 second. The fourth block embeds several join blocks within each other to create a speech bubble that displays The length of the word, the user input, is, the length of the word, and letters long for 4 seconds. Embedding the answer block in the length of block calculates the length value.

OUTPUT

Table 6-14 lists the blocks and describes the actions used in this activity.

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
ask Enter a word and wait	The sprite gets a speech bubble that displays Enter a word, opens a user input field, and waits for user input.
wait 1 secs	The script waits 1 second. No actions are performed for 1 second.
say for 4 secs	The sprite gets a speech bubble that displays the specified text or value for 4 seconds.
join	Join the two values that have been specified in the block and report the result.
join is	Join the two values that have been specified in the block and report the result.
join The length of the word	Join the two values that have been specified in the block and report the result.
answer	Hold and report the current user input value.
join letters long	Join the two values that have been specified in the block and report the result.
length of	Report how many characters the specified value contains.
answer	Hold and report the current user input value.

Activity 6-15: Pick a Random Number

This activity picks a random number between 1 and 100 and displays which number was randomly picked.

Script 6-15. Pick random number

Script 6-15 starts running when the user clicks the green flag. The next block does the rest. It creates a speech bubble, picks a random number between 1 and 100 using the pick random 1 to 100 block, and displays the phrase My favorite number is, joined with the number for 2 seconds. **Table 6-15** lists the blocks and describes the actions used in this activity.

OUTPUT

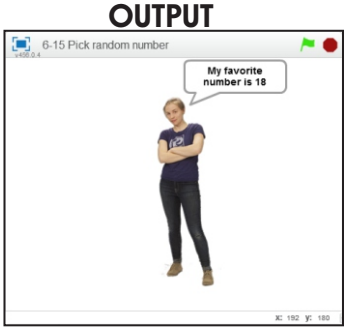
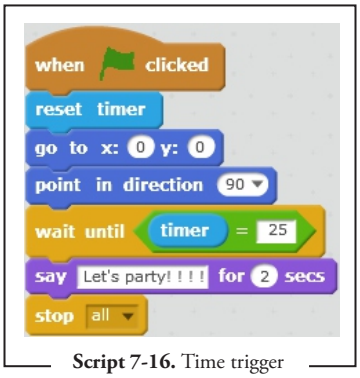


Table 6-15. Code Blocks in Pick a Random Number

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
say for 2 secs	The sprite gets a speech bubble that displays the specified text for 2 seconds.
join My favorite number is	Join the two values that have been specified in the block and report the result.
pick random 1 to 100	Pick a random value between the two specified numbers and report the result.

Activity 6-16: Timer Trigger

This activity starts the timer from 0.0 seconds. When the timer reaches 25 seconds, the script displays a text. Script 6-16 starts running when the user clicks the green flag. The next block resets the timer to 0.0 seconds. The next block moves the sprite to the center of the stage (0, 0). The next block makes the sprite face to the right. The wait until block then evaluates whether timer is equal to 25 using embedded blocks. When that condition is true, and the next actions are executed: a speech bubble displays the text Let's party!!!! for 2 seconds, then the last block stops all scripts in this project from running.



Script 7-16. Time trigger

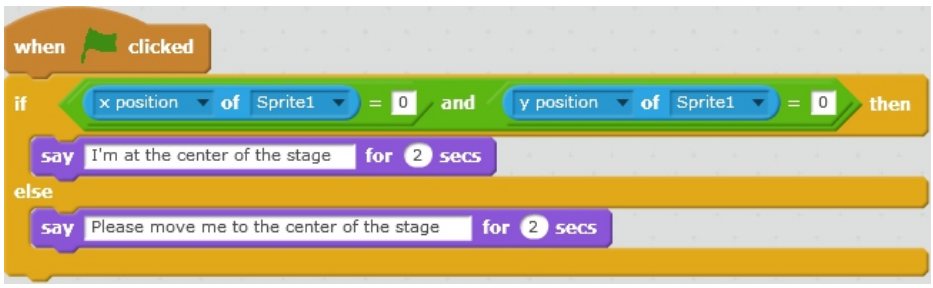
OUTPUT

Table 6-16. Code Blocks in Time Trigger

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
reset timer	Reset the timer value back to 0.0.
go to x: 0 y: 0	Move the sprite to coordinates (X = 0, Y = 0), which is the middle of the stage.
point in direction 90	Make the sprite face to the right.
wait until	Pause the script until the specified condition is true.
= 25	If the value on the left side is equal to 25, then this condition is true; otherwise, the condition is false.
timer	Hold and report the timer value.
say Let's party ! ! ! ! for 2 secs	The sprite gets a speech bubble that displays Let's party!!!! for 2 seconds.
stop all	Stop all scripts in this project.

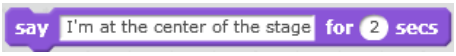
Activity 6-17: Move to the Center of the Stage

Being able to check whether your sprite has reached a specific location can be very useful to your scripts, and this script shows you how to do just that. Specifically, Script 6-17 checks whether Sprite1 is at the center of the stage (0, 0), then responds accordingly.



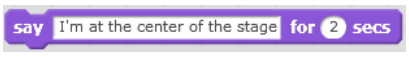
Script 6-17. Move to center

Figure 6-6. Sprite's position

Script 6-17 starts running when the user clicks the green flag. The  block then checks whether the X and Y coordinates of Sprite1 are both equal to 0. Remember the conditions tested on both sides of the  block need to be true for the whole condition to be true. A result of true means the actions in the Then portion activate: the  block creates a speech bubble and displays I'm at the center of the stage for 2 seconds. If the condition is false, action passes straight to the Else portion and a speech bubble with the text Please move me to the center of the stage displays for 2 seconds. Run the script, and move the sprite along the stage, to see the text change depending on the sprite's position. Remember, you can always see the position of the sprite the top right corner of the scripts area (see Figure 6-6). **Table 6-17** lists the blocks and describes the actions used in this activity.

OUTPUT

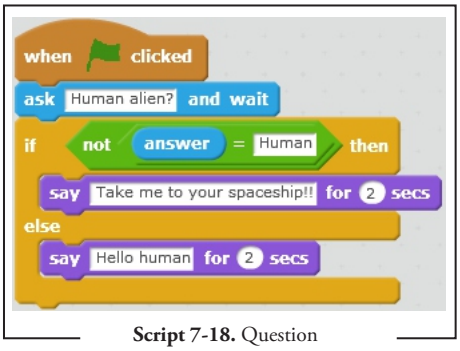
Table 6-17. Code Blocks in Move to the Center of the Stage

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	Check if the condition is true. If the condition is true, execute the actions within it, before the word else. If the condition is false, execute the actions after the word else within the block.
	Check both conditions on each side of this block. If both conditions are true, then this condition is true. Any other combination results in this condition being false.
	If the value on the left side is equal to 0, then this condition is true; otherwise, the condition is false.
	Hold and report the value of the X coordinate of the sprite.
	If the value on the left side is equal to 0, then this condition is true; otherwise, the condition is false.
	Hold and report the value of the Y coordinate of the sprite.
	The sprite gets a speech bubble that displays I'm at the center of the stage for 2 seconds.
	The sprite gets a speech bubble that displays Please move me to the center of the stage for 2 seconds.

Activity 6-18: Question

In this activity, the script asks the user to answer a question with one of two specific phrases, and then takes different actions depending on the answers. Most of the time, you'll know whether the user of your project is human or alien, but the basic technique is useful for many other types of questions.

The trigger to start Script 6-18 is the green flag. The next block creates a speech bubble and displays the question Human or alien?. This block also opens a user input field and waits for the user's input. Next, an If/Then/Else conditional statement block evaluates whether it is true that the user's answer is not Human. The answer block is snapped in the left field of = Human and the entire assembly is embedded into the not block to be evaluated. If the not block evaluates as true, then the Then action executes, creating a speech bubble that displays the text Take me to your spaceship! for 2 seconds. If the condition is false (meaning the user input does equal human), the Else portion creates a speech bubble that displays the text Hello human for 2 seconds. **Table 6-18** lists the blocks and describes the actions used in this activity.



Script 7-18. Question

OUTPUT

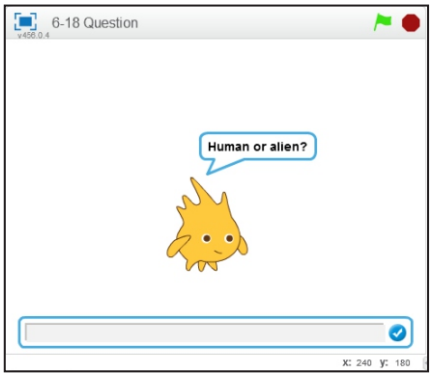


Table 6-18 Code Blocks in Question

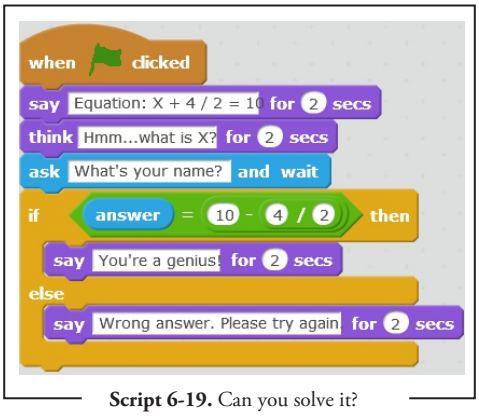
Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
ask Human or alien? and wait	The sprite gets a speech bubble that displays Human or alien?, opens a user input field, and waits for user input.
if then else	Check if the condition is true. If the condition is true, execute the actions within it, before the word else. If the condition is false, execute the actions after the word else within the block.
not	Check the condition within this block. If the condition within the block is true, return false, otherwise return true.
= Human	If the value on the left side is equal to Human, then this condition is true; otherwise, the condition is false.
answer	Hold and report the current user input value.
say Take me to your spaceship! for 2 secs	The sprite gets a speech bubble that displays Take me to your spaceship! for 2 seconds.
say Hello human for 2 secs	The sprite gets a speech bubble that displays Hello human for 2 seconds.

Activity 6-19: Can You Solve It?

This script displays a problem for the user to solve. The user needs to solve for X in the equation X + 4 / 2 = 10. The text response that the script displays depends on if the answer is right or wrong.

Script 6-19 also starts running when the user clicks the green flag. The next block creates a speech bubble and displays the text Equation: X + 4 / 2 = 10 for 2 seconds. The next block creates a thought bubble and displays the text Hmm…what is X? for 2 seconds. The 

block creates a speech bubble that displays What's your answer?, opens a user input field, and waits for the user's input. Next is an If/Then/Else conditional statement that evaluates whether the user input equals the result of 10-4/2. If the condition is true, the block in the Then portion creates a speech bubble and displays the text You're a genius! for 2 seconds. If the condition is false, the block in the Else portion block creates a speech bubble that displays the text Wrong answer. Please try again. for 2 seconds. **Table 6-19** lists the blocks and describes the actions used in this activity.



Script 6-19. Can you solve it?

OUTPUT

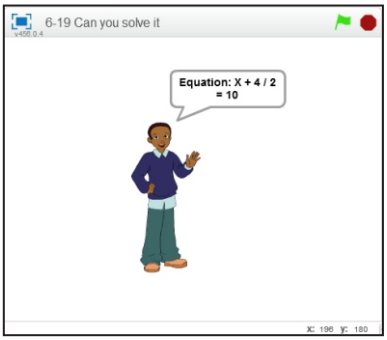
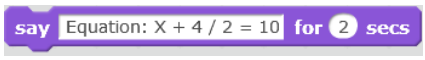
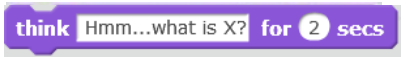
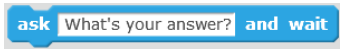
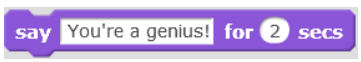
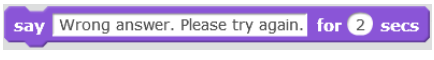


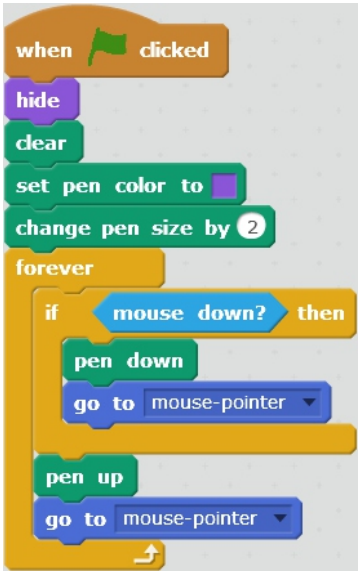
Table 6-19. Code Blocks in Can You Solve It?

Blocks	Actions
	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
	The sprite gets a speech bubble that displays Equation: X + 4 / 2 = 10 for 2 seconds.
	The sprite gets a speech bubble that displays Hmm…what is X? for 2 seconds.
	The sprite gets a speech bubble that displays What's your answer?, opens a user input field, and waits for user input.
	Check if the condition is true. If the condition is true, execute the actions within it, before the word else. If the condition is false, execute the actions after the word else within the block.
	Check if both sides of the equal sign are equal. If both values are equal, then this condition is true; otherwise, this condition is false.
	Hold and report the current user input value.
	Subtract one value from another and report the result.
	Divide one value by another (4/2) and report the result.
	The sprite gets a speech bubble that displays You're a genius! for 2 seconds.
	The sprite gets a speech bubble that displays Wrong answer. Please try again. for 2 seconds.

Activity 6-20: Drawing with the Mouse-Pointer

This script lets you draw in the stage area. To draw, you need to click the left mouse button, hold it down, and move it around the stage. If the primary mouse button or left mouse button is not pressed then the pen will not draw because the pen up block is in effect. Script 6-20 starts running when the user clicks the green flag. The next block makes the sprite disappear from the stage area. The next block clears the stage area of any marks made by the pen or stamp. The next pair of blocks sets the pen color and changes the pen size. Next is a C block that repeats the sequence of actions within it forever block.

First within this sequence is the if then block, which checks whether the left mouse button is pressed using the embedded mouse down? block. If this is true, the sequence within the conditional statement block is executed. The first block places the pen on the stage, ready to draw. The next block moves the pen toward the mouse-pointer. You can use your mouse-pointer to draw in the stage area, because as long as the left mouse button is held down, everywhere the mouse-pointer moves, the pen draws a trail. When the mouse button is no longer pressed, the script moves out of the if then block and the next block lifts the pen from the stage. The last block also moves the pen toward the mouse-pointer, but no drawing is visible because the pen is lifted. **Table 6-20** lists the blocks and describes the actions used in this activity.



Script 6-20. Drawing

OUTPUT

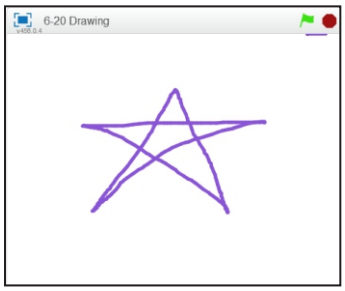


Table 6-20. Code Blocks in Drawing with the Mouse-Pointer

Blocks	Actions
when clicked	Clicking the green flag activates the script. The green flag is the trigger to start the script running.
hide	Make the sprite disappear from the stage.
clear	Remove all marks previously made by the pen or stamp.
set pen color to	Set the pen color to purple color.
change pen size by 2	Change the pen size to 2.
forever	Repeat all the instructions/blocks within this block forever.
if then	Check if the condition is true. If the condition is true, execute the actions within it. If the condition is false, skip to the next block.
mouse down?	Check if the primary mouse button is clicked. If the primary mouse button is clicked, then the condition is true. If not, then the condition is false.
pen down	The pen is on the stage and is ready to draw.
go to mouse-pointer	Move the sprite toward the mouse-pointer.
pen up	The pen is off the stage and cannot draw.
go to mouse-pointer	Move the sprite toward the mouse-pointer.

Summary

In this chapter, you learned about some advanced computer programming concepts, including the following:

- Conditional statements
- Conditions
- Loops
- User input

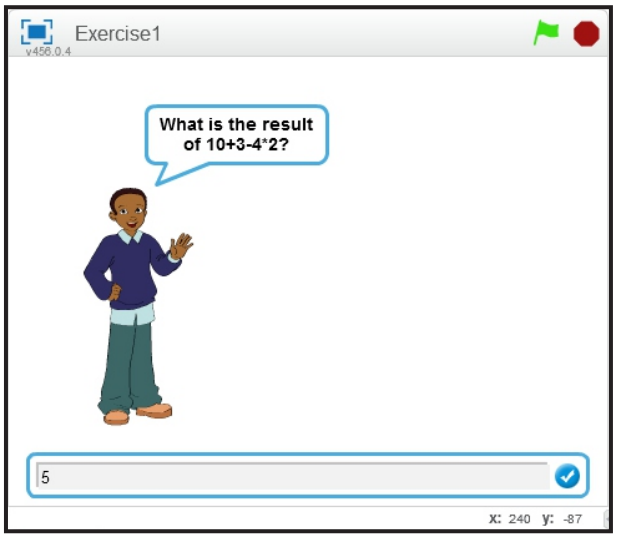
All programming languages rely on these concepts. The blocks that you need to execute these concepts in Scratch are available in the **Control, Operators,** and **Sensing** categories. With them, you can create dynamic scripts that can produce a different result every time they're run, depending on the condition and/or user input. In the next chapter, you'll investigate another advanced concept: variables, which are items that hold a value that can change. They work similar to the way that the answer block saves user input.

Snap Script

a Short hands-on activity

1. Create a script that asks the user for the answer to the equation 10 + 3 – 4 * 2. Use an If/Then/Else conditional statement block to display text depending on if the answer is right or wrong.

Output:



2. Create a script that displays the result of each of the following equations: 2 + 3, 10 – 7, 13 * 4, and 13 / 4. Use join blocks.

Output: