

Learning goals: algorithms, invoking/calling/using methods or functions, program

Cut scene: The interactive introduction at the beginning (called a cut-scene), introduces the idea of an algorithm as a step-by-step sequence of commands to solve a problem. There is a correct order in which the commands need to be executed in order to solve the problem.

It also introduces the idea of “commands” as instructions. Commands are written with no spaces between the words and with a trailing set of parentheses, these commands or functions or methods give the Byte character his suite of possible instructions.

For example, Byte currently has the behaviors:

```
moveForward()  
collectGem()
```

Before you start:

- Have students swipe left and right on the left side of the screen to see that Byte’s world rotates.
- Have students READ the instructions! There is often important information in the instructions that will save time and frustration for the students in the written instructions.
- Tap any red words in the instructions. These will define any new and important terms. In this lesson, the word **command** is linked and a definition is provided.
- Note the commands on the bottom of the screen to the students: `moveForward()` and `collectGem()` should be all the commands in that are needed in this program. Later on, we will use a keyboard to type commands ourselves, but for these first few lessons, students should get used to tapping the shortcuts listed at the bottom.

Activity: Have students write their code by tapping on the commands in the correct order and test it by tapping the *Run* button to make Byte perform their algorithm. It’s OK to make mistakes. Students can fix their mistakes and press *Run* again to test their program.

Give students sufficient time to explore this playground but students should not move ahead to the next activity until it is introduced. (See solution below)

Follow-up activities:

- To students: “You just wrote a program!” “Does anyone want to share the statements they wrote in their program?” “Did your program work?” “How do you know? (Byte dances!)”
- Have students modify their program so that it does not solve the problem. I.e., put a mistake in it. What does Byte do?
- Instructor can demonstrate a solution with a mistake and show how to use the backspace and return shortcuts (provided on the bottom-right of the playground).

- Before leaving this lesson, have the students tap the red Swift definition in the Congratulations pop up window to learn about Swift.

Solution:

The screenshot displays the 'Issuing Commands' lesson interface. On the left, a code editor shows the following Swift code:

```
moveForward()  
moveForward()  
moveForward()  
  
collectGem()
```

A tooltip titled 'command' defines it as 'Code that tells an application to perform a specific action.' Below the code, a list of instructions is visible:

- 1 Look for the...
- 2 Enter the correct combination of the `moveForward()` and `collectGem()` commands.
- 3 Tap Run My Code.

On the right, a 3D game environment shows a yellow robot character on a platform. A 'Congratulations!' pop-up message is displayed, stating: 'You've written your first lines of Swift code. Byte performed the commands you wrote and did exactly what you asked, in exactly the order that you specified.' Below the message is a 'Next Page' link. The bottom of the screen features a command palette with 'collectGem()' and 'moveForward()' buttons, and navigation icons.