Name_____

Lesson 6: Worksheet 6.3 – Design your own function

In this activity, you will design your own function and use it to write a program for Edison.

What, exactly, are functions?

Now that you have been programming for a while with Edison and EdPy, you have used a range of different functions from the EdPy library.

As you know, a function is a piece of code that performs a particular role or job in the program depending on which input parameters are used.

But you might not have realised that all of the functions you have used so far have actually executed multiple lines of code when run by the program.

That's because a function is a block of organised, reusable code that is used to perform a single, related action.

Functions are very helpful because they allow us to program is a more modular way, using the same block of code at various points throughout the program. To get your program to run all of those lines of code you only need to type one line: calling that function.

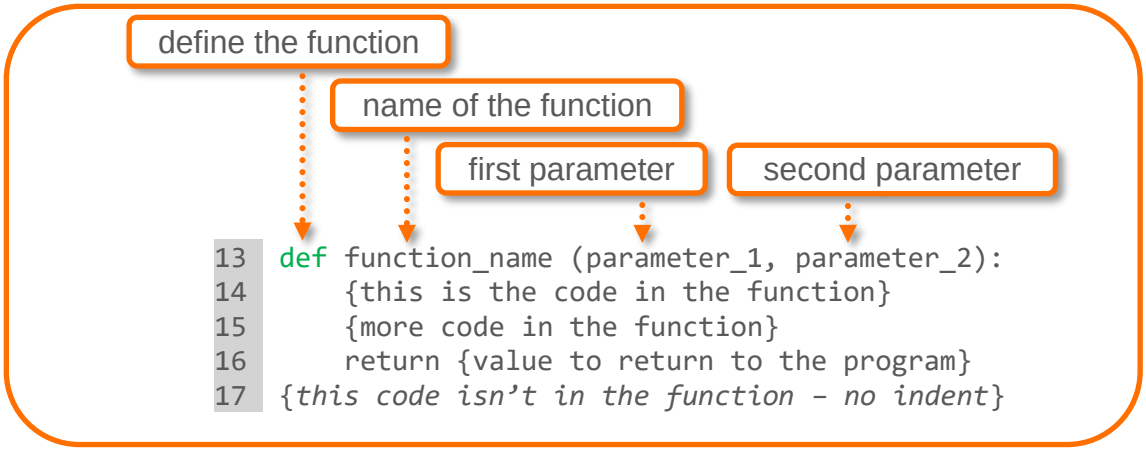
Functions make reusing code easier and more efficient when we program.

> Watch a short video by code.org explaining why functions are so useful in programming: https://youtu.be/8T5acEwfJbw

Making your own function

So far, every time you have used a function in EdPy you have called that function from the Ed library. You can also make your own functions.

In Python, functions look like this:



Anything indented is a part of the function. Anything not indented is not a part of the function, but the next line of code in the program.

It's important to note that the functions you create are no different from the functions you've used from the Ed library so far.

Name_____

When you call your function (with or without parameters), the program jumps from the call to the code in the function (the indented code). The program then runs this code before returning to the line where you made the call.

If you set your function to return a value, when the program returns to the line where you made the call, the function call gets replaced by the value that the function has returned.

This is important because the program will always resolve functions, then maths orders then expressions in this order.

Organising your code

The convention in EdPy is to write the definition of your functions at the end of your code after you have already used your function in your program.

This is so your code is nice and neat. By organising your code this way, all your functions, whether you are using your own or calling functions from a library, can be written in the main part of your program in a visually clean, organised way. Your function definitions can sit outside of this, further down at the bottom of the program.

Your turn:

Task 1: Practice defining a function

Look at the following program:

```
1
2  #-------------Setup----------------
3
4  import Ed
5
6  Ed.EdisonVersion = Ed.V2
7
8  Ed.DistanceUnits = Ed.CM
9  Ed.Tempo = Ed.TEMPO_MEDIUM
10
11 #--------Your code below-----------
12 directionToMove=Ed.SPIN_LEFT
13 speedToMoveAt=Ed.SPEED_8
14 distanceToMove=360
15
16 #call the function
17 moveOnClap(directionToMove, speedToMoveAt, distanceToMove)
18
19
20 #user defined functions
21 def moveOnClap(direction, speed, distance):
22     Ed.ReadClapSensor()
23     while Ed.ReadClapSensor()==Ed.CLAP_NOT_DETECTED:
24         pass
25     Ed.Drive(direction, speed, distance)
26
27
```

Name_____

Line 17 in the program is the function call. Lines 21 to 25 are defining the function. Remember, the convention is to define your function at the bottom of your program to keep everything neat and organised.

Write the program and download it to your Edison robot. Run the program to see how it works.

1. We could write a function to make the Edison robot drive in a square shape. Fill in the missing words in the function below to finish writing this function.

```
def driveInaSquare():

    for i in range(___):

        Ed.Drive(_____________,_____________,_____________)

        Ed.Drive(_____________,_____________,_____________)
```

2. Write the syntax for the code to call this function. In other words, what would you type in your program to call this function?

_____

Write a program that will drive Edison in multiple squares. You will need to define the 'driveInaSquare' function, and your program will need to call this function more than once. Download and run the program to see how it works.

3. Does the program do what you expected it to do? If not, describe what you expected and what the program actually did. Describe any issues you had getting your program to work.

Task 2: Design your own function

Write your own function which will have Edison do something when you clap. You could make Edison dance, flash an LED, turn in a circle or anything else you like!

Name_____

Step 1: Design

First, write a flowchart which summarises your program graphically. Either make your flowchart by drawing it out on paper or use a program such as Google Slides. Be sure to use the correct shapes in your flowchart to represent the program's start, processes, decision points and flow.

The shapes you will need will depend on your flowchart. At a minimum, you will need the oval start shape, the rectangle process shape and the diamond decision shape in your flowchart.

Step 2: Code

Translate your idea into code in the EdPy app. Use your flowchart as your guide and code your function to include each step you laid out in your flowchart.

Step 3: Test

Write a test program that includes your function and additional code to 'exercise' your function. (Exercising a function means calling it in your code.) See if your function works as you planned and expected it to work. If not, revisit your flowchart and code, adjusting as needed. Experiment to see what works!

4. Describe what your function did.

5. Describe any problems you had.