# Visual to text coding LESSON 10: **Simple Functions**

Please refer to the online lesson plan on the DT Hub to access all website links and additional resources.

This is the tenth in a series of lessons to transition from visual coding to text-based coding with a General Purpose Programming language.

Included videos can be used by a beginner teacher and/or students to see how to code each of the simple programs step-by-step in *all three languages*: Scratch, Python and JavaScript.

This lesson may take two to three 45-minute periods. It introduces the coding concept of functions. Functions can help organise code, reduce repetition and more to be explored later.

# Curriculum links

Links with Digital Technologies Curriculum Area

| **Strand** | **Year** | **Content Description** |
| --- | --- | --- |
| Processes and Production Skills | Year 5-6 | Design algorithms involving multiple alternatives (branching) and iteration (AC9TDI6P02) . |
| Year 7-8 | Design algorithms involving nested control structures and represent them using flowcharts and pseudocode (AC9TDI8P05) .  Trace algorithms to predict output for a given input and to identify errors (AC9TDI8P06) . |

# Assessment

Students can undertake a self-reflection of the programming task. The teacher can use the completed self-assessments to assist in summative assessment.

* Download the self-assessment sheet in Word or PDF format.

In assessing code in languages like Python or JavaScript, consider a rubric that brings in important skills for General Purpose Programming.

* Download a sample rubric in Word or PDF format.

# Learning hook



Wikimedia Commons

Imagine you’ve been put in charge of a colony ship so that civilisation can be built on a new planet. Does this task seem daunting?

*As a class, see if you can list some of the tasks that will need to be done once the ship lands on the new world.*

When projects are large, it’s usually necessary to delegate and rely on different people with different experience, talent and skills. Did you know that, according to Wikipedia’s statistics in February 2020, approximately 140 000 people performed an action or contributed to Wikipedia in the previous 30 days?

As computer programs get larger, it becomes impractical to have all the different tasks in one large program. Instead, we separate the code for specific tasks into smaller **functions** (sometimes called subroutines).

The code left behind is now called the **main program** (or main routine). From there we start the program, activate(or **call**) the functions we want to use and then end the program.

This has several advantages:

* Instead of just thinking about the sequence of statements in a program, you can now organise your code in terms of the *actions* your program does.
* Functions can make your code more *readable*, especially if they are named well. It’s easier to find the bits of the code you want to improve or fix, and your main program is smaller and cleaner.
* It’s easier for *multiple programmers* to work on the same project. One person can concentrate on the code inside a function while another works on the main program.
* Functions can be written to *adapt* their behaviour according to values provided. We’ll learn how in Lesson 11.

# Learning map and outcomes

In this lesson, students will:

1. access an online programming environment for visual code (**Scratch**) *and* for General Purpose Programming (**Python** or **JavaScript**),
2. code functions to draw shapes, then reuse them to make fun patterns,
3. use functions to draw a cosmic scene full of stars.

# Learning input

First, have students prepare their Internet browser to use **Scratch** along with an online environment for **Python** or an online environment for **JavaScript**. See this video for how to access and test each of these environments. *Note that this JavaScript environment does not work in Microsoft Edge (last tested February 2020).*

SIDEBAR – What’s with the turtle?

Turtle graphics has been around since the original Logo language for teaching programming to students, developed in 1966-1967. It is being introduced at this point in this course because it is an effective context to learn about functions, but examples will also be provided for writing functions outside of the turtle graphics context.

Like today’s Bee-bots and other rovers used in classrooms, the ‘turtle’ can be thought of as a robot moving about on the floor, and capable of drawing lines as it moves. Programmers issue simple instructions like **forward 20** to move forward 20 units of distance, or **right 90** to rotate clockwise 90 degrees.

To activate the pen, the typical command is **pen down** (think of a pen being dragged along the floor by the robot). The command **pen up** lifts the pen off the floor, allowing the robot to move around without drawing.

In Python and JavaScript environments with turtle graphics, these commands can be typed and run within your programs just like other lines of code.

In Scratch, sprites are already moved around using the blue motion blocks. Any sprite can become a drawing turtle by using the Pen extension, which brings in the **pen down** and **pen up** commands.

Individually, practice writing a simple program in Scratch to draw the alphabet letters that make your initials, then code and test the same program in Python or JavaScript.

Notes:

* No need to create functions yet.
* Try to make your letters as accurate as possible, with the same height and appropriate width.
* It is easier at first to do straight lines than curves, so consider doing letters that are “blocky”.

*Sample solution code for the initials SPJ:* *Scratch, Python, JavaScript*

# Learning construction

**STEP 1: FUNCTION CHALLENGE IN SCRATCH**

Watch this video that demonstrates a basic shape-drawing function in Scratch and ends with a challenge. Try it yourself before checking the conclusion in the solution video.

*Final code: Scratch*

**STEP 2: FUNCTION CHALLENGE IN PYTHON AND JAVASCRIPT**

Watch this video that shows how to implement STEP 1 in Python and JavaScript. Try it yourself.

*Solution code: Python, JavaScript*

**STEP 3: FUNCTION EXERCISE WITH TURTLE GRAPHICS**

Carefully read the pseudocode below.

1 BEGIN

2 Function drawHouse()

3Forward 50

4 Left 90

5 Forward 50

6 Left 45

7 Forward 35

8 Left 90

9 Forward 35

10 Left 45

11 Forward 50

12 Left 90

13 EndFunction

14

15 Pen down

16 drawHouse()

17 Pen up

18 Forward 100

19 Pen down

20 drawHouse()

21 END

Now, fill in the missing words in the paragraph below, to explain what the program does.

**This program draws not one but \_\_\_\_\_\_\_ houses side by side. The main program begins at line number \_\_\_\_. After activating the pen, the program calls the \_\_\_\_\_\_\_\_\_\_\_\_\_\_ function, which produces a shape. The function begins at line number \_\_\_\_ and ends at line number \_\_\_\_. Once the house shape is drawn, we return to the \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_. Next, the pen is \_\_\_\_\_\_\_\_\_\_\_\_\_\_ and the turtle moves forward 100 units. Finally, the pen is reactivated and \_\_\_\_\_\_\_\_\_\_\_\_\_\_ is called again.**

**Step 3 solution:**

This program draws not one but two houses side by side. The main program begins at line number 15. After activating the pen, the program calls the drawHouse function, which produces a shape. The function begins at line number 2 and ends at line number 13. Once the house shape is drawn, we return to the main program. Next, the pen is deactivated and the turtle moves forward 100 units. Finally, the pen is reactivated and drawHouse is called again.

Lastly, implement the code in Scratch, as well as Python *or* JavaScript.

*Solution code: Scratch, Python, JavaScript*

**STEP 4: FUNCTION EXERCISE WITHOUT TURTLE GRAPHICS**

Carefully read the pseudocode below. This program does not use turtle graphics.

1 BEGIN

2 Function assignPasscode()

3 **answer** ← Input 'What is your name?'

4 Display 'Hi', **answer**, '. Here is your passcode:'

5 **passcode** ← Random number between 1000 and 9999

6 Display **passcode**

7 EndFunction

8

9 Repeat 4 times

10 Display 'Assigning a new passcode...'

11 assignPasscode()

12 Display ''

13 End Repeat

14 END

Now, fill in the missing words in the paragraph below, to explain what the program does.

**This program assigns random passcodes for four people. The main program begins at line number \_\_\_. It contains a \_\_\_\_\_\_\_ which happens \_\_\_ times. Each time, it prints ‘Assigning a new passcode...’ and also \_\_\_\_\_\_\_\_ a function called assignPasscode(). The function asks the user for their \_\_\_\_\_\_\_\_. It then generates a passcode as a number between \_\_\_\_\_\_\_ and \_\_\_\_\_\_\_. This is displayed for the user.**

Step 4 solution:

This program assigns random passcodes for four people. The main program begins at line number 9. It contains a loop which happens 4 times. Each time, it prints ‘Assigning a new passcode...’ and also calls a function called assignPasscode(). The function asks the user for their name. It then generates a passcode as a number between 1000 and 9999. This is displayed for the user.

Lastly, implement the code in Python *or* JavaScript:

* If doing Python, use a regular Python project in repl.it as in Lessons 1 to 9.
* If doing JavaScript, use JSFiddle as in Lessons 1 to 9.

*Solution code: Python, JavaScript*

# Challenge

These challenges use the skills covered so far.

1. Write and test a function to create a small star shape. In the same program, write and test another to create a big star shape. See the images below for sample shapes to aim for.

 A picture containing object

Description automatically generated

With your functions created, use the main program to create a star-filled sky with a mix of large and small stars.

*Note: You may choose to use loops in the main program to help reduce repetition.*

*Solution code: Scratch, Python, JavaScript*

1. Write and test a function to create a small geometric shape. Use that function as part of a main program to create a geometric design. As a further challenge add colour to the design.
2. (Without turtle graphics.) Choose a popular song with a few versus as well as a chorus that recurs. Write and test one function for each verse, as well as one function for the chorus. These functions should simply display the words.  
     
   In the main program, call the verse and chorus functions in the correct order to display the complete song. You may need to call the chorus function more than once.

# Resources

* Setting up online environments
* Online environments for coding in each language
  + Scratch
  + repl.it : an online environment suited to Python
  + JSFiddle : an online environment suited to JavaScript
* Cheat sheets listing basic commands for coding:
  + Python Cheatsheet (from Grok Learning)
  + JavaScript CheatSheet (Tip: Press the little blue tabs to move Variables, Basics, Strings and Data Types to the top.)