Hi! In this video I will show you how to build your own functions in Python.

[SLIDE 2] Let’s start by remembering how you build your own functions in C.

[SLIDE 3] In the first line you declare your function. That is,

[SLIDE 4] you define the data type of the value returned by your function,

[SLIDE 5] the name of your function and

[SLIDE 6] the type and name of the input arguments.

[SLIDE 7] So, in this simple example function, the type returned by the function is an integer value, the function is called sum and it receives two input arguments named a and b, both of the integer type.

[SLIDE 8] Now, in Python this first line is simpler, as you don’t need to define types.

[SLIDE 9] The first word, def, does not define a type. It is a reserved word to signal the start of the definition of a function.

[SLIDE 10] Like C, the second word is the name of the function and

[SLIDE 11] unlike C, you only need to name the input arguments. You don’t need to define their types.

[SLIDE 12] Finally, the first line of a function in Python must end with a colon (:), to signal the start of a block of code which are the instructions of the function.

[SLIDE 13] Coming back to the example of the function sum, this is how the first line of this function looks like in Python.

[SLIDE 14] Looking at the body of the function, the main differences is in Python using indentation to identify the instructions inside the function whilst C uses the curly brackets.

[SLIDE 15] Finally, a big difference is that C only allows you to return a single value, but in Python

[SLIDE 16] a function can return several values.

[SLIDE 17] To finish this introduction to functions in Python, let’s have a look at how they are included in a programme. This will also allow me to show you the differences between the structure of a C programme and a Python programme.

[SLIDE 18] Here, you can see a C programme where 2 functions called

[SLIDE 19] circle area and

[SLIDE 20] circle perimeter have been declared. They take as input argument the radious of a circle and return the area and the perimeter of the circle, respectively.

[SLIDE 21] On the right of this slide you can see the equivalent in Python. In this case, I have chosen to write a single function that calculates the area AND the perimeter and returns these two values. So, instead of writing two different functions, I only wrote one.

Please, notice that you can also write two different functions, one to calculate the area and another to calculate the perimeter.

[SLIDE 22] Now, let’s have a look at the main function in C.

[SLIDE 23] This block of code is in charge of creating the necessary variables and getting data from the user.

[SLIDE 24] Next, the functions circle_area and circle_perimeter are called and their results are stored in the variables area and perimeter, respectively.

[SLIDE 25] Finally, two messages are printed on screen informing the user about the values of the area and perimeter of the circle.

[SLIDE 26] Let's now look at how the main part of the programme looks in Python.

In Python, there is no need for a main function. The body of the programme is simply written after all functions have been declared.

[SLIDE 27] The first two lines in the Python code perform the same task than the first 3 lines in C. As you don't need to declare variables in Python, you only need one line to print the message on screen and another to store the data entered by the user in the variable radius.

[SLIDE 28] The third line in the Python code performs the same tasks that lines 4 and 5 in the main function of the C code. Thanks to the fact that Python functions can return more than 1 value, we can simply list the variables where want to store the returned values (variables area and perimeter in this example) to the left of the assignment operator. To the right, we call the function.

[SLIDE 29] Finally, the last 2 lines in the Python code are in charge of printing the message with the values of the area and the perimeter of the circle.

[SLIDE 30] Notice that we must use the function str, that transforms the numerical values of the variables area and perimeter to a string type, to be able to print the values on screen.

[SLIDE 31] With this, we finish our work understanding how the instructions your learnt last year in C are translated into Python.

In the next video you will learn about how Python deals with errors, a very useful feature.