Hi! In this video I will show you how to use loops in Python, starting from what you know in C.

[SLIDE 2] First of all, Python only has **while** loops and **for** loops. The do-while loop you can use in C does not exist in Python.

[SLIDE 3] The while loop is built exactly in the same way it is built in C, except for the differences in syntaxis. Let's look first at the similarities.

[SLIDE 4] First, both loops need a control variable that is initialised before checking the condition of the while loop. In this case, the variable controlling the loop (that is, the variable whose value determines whether the instructions inside the loop are executed or not) is variable i, that gets initialised to the value 0.

[SLIDE 5] Next, both loops check the condition to keep executing the instructions inside the loop. In this example, this is the condition "i is lower than 5". If the condition is true, the instructions inside the loop are executed. Otherwise, the loop ends.

[SLIDE 6] Also, both loops update the control variable in such a way that its value gets closer to the point where the condition of the while loop will become false.

[SLIDE 7] Finally, both loops have instructions to execute in case the condition of the loop is true.

As you can see, the structure of the while loops in Python and C is exactly the same. The differences are due to the different syntaxis of the languages. Let's have a look at them.

[SLIDE 8] First, Python doesn't require semicolons to signal the end of an instruction. Every instruction in Python must be written in a different line.

[SLIDE 9] Second, in Python we do not use curly brackets to signal the start and the end of the block of instructions inside the while loop. To signal that start of the block of instructions of the while loop we use a colon and

[SLIDE 10] to identify the instructions inside that block, we use indentation.

There are also differences in how the function used to display information on screen is named and used, but we will see that in the next video.

Ok, that's all for the while loop for now. Let's move now to the for loop.

[SLIDE 11] In Python, the for loop is used to iterate over a sequence only. For example, iterating over a string, visiting every character in it.

[SLIDE 12] Or iterating over a range of numbers, visiting every number in the range.

The for loops in C can also be used to iterate over a sequence, but they can also replicate any while loop. That's not the case of the for loops in Python. For this reason, in the following we will see only examples of the for loop as an iterator.

[SLIDE 13] Here, I am showing you an example of a piece of code – written in C and in Python - that uses a for loop to iterate over the content of a string. In each iteration it prints the corresponding character on screen. Let's see how this is done.

[SLIDE 14] First, both pieces of code store the word "Programming" in variable x.

[SLIDE 15] Next, in C you need to declare the variable c that you will use as the control variable of the for loop. This is not necessary in the Python code.

[SLIDE 16] Now, we get to the for loop acting as an iterator over the string. That is, visiting every character in the string. While in C you need to define the starting position, the condition to end the iteration and how the control variable needs to be updated, in Python you simply instruct variable c "to traverse" the string. You do that by writing "for c in x". That's all.

[SLIDE 17] Finally, both pieces of code print the corresponding character on screen.

[SLIDE 18] Here you can see a more common example of the use of the for loop in Python. Here, both pieces of code print the sequence of numbers from 0 to 5 on screen.

[SLIDE 19] To do so, in C you first need to declare the control variable used in the loop. You don't need to do so in Python.

[SLIDE 20] Next, both pieces of code iterate over a sequence of numbers. In C, this is done by defining the starting value, the condition to stop the iteration and the increment of the control variable.

In Python, you simply write "for i in range(6)".

[SLIDE 21] The function range produces a sequence of numbers that, by default, start at 0 and the increments are done in steps of one unit. The last number produced by the sequence, is N-1 if the single input argument is N. Thus, by having range(6) the sequence of numbers produced is 0, 1, 2, 3, 4 and 5.

[SLIDE 22] If you do not want to use the values by default, you need to specify the starting value and the increment when calling the function range in the way described here: the first input argument is the start number of the sequence, the second -N- is the number where the sequence stops (so it shows all numbers from the starting point to N-1) and the third input argument is the increment.

[SLIDE 23] Thus, by writing range(0,6,1) you obtain the same behavior than writing range(6).

OK, with this, I have finished the basics of for loops in Python.

[SLIDE 24] In the next video, we will look at the two functions that allow you to display information on screen and obtain information from the user using the keyboard.