Hi! In this video we will learn how the instructions you learnt in C can be translated to Python. To do so, we will first remember the main instructions in C to then learn how they are transformed into Python instructions.

[SLIDE 2] In blue, you can see the 3 types of instructions you learnt in C:

[SLIDE 3] First, you can see instructions that allow you to create new variables, such as:

- int x to create an integer variable called x;
- float y to create a variable called y that can store a decimal value;
- char a to create a variable called a that will be able to store a character and
- the special variables that allow us to store memory addresses, also known as pointers. In this example, int *p is creating a variable called p that stores the memory address of an integer value.

[SLIDE 4] Second, we have instructions that use operators to manipulate the content of variables. For example:

- the assignment operator, that allows us to store a specific value in a variable or
- arithmetic operators, that allow us to update the content of variables by performing arithmetic operations. Here, I am just showing the operator used to sum numbers, but of course, there are more.

[SLIDE 5] Finally, we have flow control instructions that modify the sequential execution of instructions. They can be:

[SLIDE 6] conditional statements or loops.

[SLIDE 7] In the case of conditional statements, we have the if-else instructions. An if-else instruction checks whether a condition is true. In the case of the example shown in the slide, the condition is “x equals 3”. If this condition is true, the instruction “make x equal to x multiplied by 2” is executed. Otherwise, the instruction “make x equal to x minus 1” is executed.

[SLIDE 8] In C, we also have switch statements, that correspond to a shortened version of a series of nested if-else instructions or a sequence of if instructions. In the example, the switch-case statement performs the same task as the if-else instruction above.

[SLIDE 9] In terms of loops, C has 3 types of loops:

[SLIDE 10] the while loop

[SLIDE 11] the for loop, which is a shortened way of writing a while loop. On the left lower part I have written an example of a while loop and its equivalent for loop. Finally, C also has

[SLIDE 12] the do-while loop. These are all the types of instructions you can execute in C and almost all of them have an equivalent in Python.

[SLIDE 13] Now, in C all these instructions are organized in functions.

[SLIDE 14] You can have functions already available in the C libraries, like the function printf, or

[SLIDE 15} you can build your own function, like the one in the example that returns the double of the value entered as input argument.

[SLIDE 16] OK, with this overview of the instructions you used for your C programmes, we will now see how these instructions are written in Python.

[SLIDE 17] Let’s start with the declaration of variables. In C, before using a variable you need to declare its type.

[SLIDE 18] In Python, variables do not need to be declared. When you assign a value to a variable, then the variable is created and the type is automatically defined. So, this step is not required in Python.

[SLIDE 19] These are some of the built-in types in Python. This list of types is not exhaustive, but it includes the most commonly used types.

[SLIDE 20] Python has a built-in string type, this makes the work with strings much easier than in C since you don't need to manipulate the individual characters inside an array. Characters are simply treated as strings of length 1. So, in this example, when you assign the string Ben to the variable name, that variable gets created (if it didn't exist before) and the type string is associated to it.

[SLIDE 21] Integer and decimal numbers also have their own types, as in C.

[SLIDE 22] Lists are the way Python deals with collections of numbers. Lists are more flexible than arrays. We will see the details about lists in Week 2.

[SLIDE 23] Finally, you can also use Boolean variables in Python. OK's that's all regarding variables for now. Let's move to the operators.

[SLIDE 24] This is a non-exhaustive list of operators in Python. You can check the details in the link at the bottom of the slide. I have highlighted in red the operators that are different from C.

[SLIDE 25] The first one is the exponentiation operator, that doesn't exist in C. Now, instead of writing your own function to perform this operation, you can simply use this operator.

[SLIDE 26] The second one is the integer division. The normal division in Python will return an integer or decimal number depending on the types of the numbers you are dividing. If you want to trunk the resulting number, then you use this operator.

[SLIDE 27] Finally, the logical operators in C are replaced by their names in Python.

[SLIDE 28] Let's move now to the flow control instructions.

[SLIDE 29] Let's start with the conditional statements.

Python only has the if-else instruction. The switch-case statement available in C does not exist in Python. In Python, the if-else instruction can be written in different ways. Here, I will show you the one most similar to what you already know from C.

[SLIDE 30] Here you can see the same if-else instruction written in C and in Python. Let's have a look at 2 important differences:

[SLIDE 31] First, please notice that you need to add a colon (the symbol made of two points in vertical position) immediately after the condition and the else word.

[SLIDE 32] Second, notice we don't need to use curly brackets to signal where the instructions associated to the if-else start and end. To do so, we use indentation. That is, we write the instructions inside the if-else indented to the right. In this way, indentation in Python is used to signify code blocks in the way curly brackets do in C. As a result, all consecutive instructions indented to the same distance to the right, belong to the same code block.

[SLIDE 33] In this example, you can see that indentation is used to signal what instructions must be executed inside the if and inside the else. The instruction not indented here, it is outside the if-else statement.

[SLIDE 34] Ok, that's for now. In the next video we will see how to use one of many developing environments available for Python to them jump to loops.