Python Bootcamp Day-3

Loops in Python

👩🏻‍💻

A loop is a sequence of instructions that is continually repeated until a certain condition is reached.

Python provides three ways for executing the loops - while, for, and nested loops.

While Loop-

Repeats a statement or group of statements while a given condition is TRUE. It tests the condition before executing the loop body.

The while loop accepts a single expression and the loop continues to iterate as long as the test continues to return true.

```
while Expression: Block1 else: Block2
```

```
x=0 while(x<5): print(x) x=x+1 #OUTPUT # 0 # 1 # 2 # 3 # 4
```

For Loop-

Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.

for

```
for TARGET in OBJECT: BLOCK else: BLOCK
for x in [1,2,3,4,5]: print('I can count number',x) #Output #I can count
number 1 #I can count number 2 #I can count number 3 #I can count number
4 #I can count number 5
```

for each iteration of the loop, x is set to the next value within the array producing the output.

Nested Loop-

You can use one or more loop inside any another while, for.

Try it yourself 😎

Loop Control Statements

👩🏻‍💻

Loop control statements are used to change the order of execution of the code, to terminate the execution or loop manually for any specific conditions(by using break), or even to write a conditional statement without adding any block of code inside it(by using pass)

It changes execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed.

Python supports the following control statements- Break, Continue, and Pass

Let's see what it is all about 😇

Break statement

The `break` statement is used to terminate the loop or statement in which it is present. After that, the control will pass to the statements that are present after the break statement, if available. If the break statement is present in the nested loop, then it terminates only those loops which contains `break` statement.

Syntax:

break

Continue statement

`Continue` is also a loop control statement just like the break statement. `continue` statement is opposite to that of break statement, instead of terminating the loop, it forces to execute the next iteration of the loop.

Syntax:

continue

Pass statement

The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute. It is like `null` operation, as nothing will happen is it is executed. `Pass` statement can also be used for writing empty loops. Pass is also used for empty control statement, function and classes.

Syntax:

```
pass
```

Conditional Statements

👩🏻‍💻

Conditional Statement in Python perform different computations or actions depending on whether a specific Boolean constraint evaluates to true or false. Conditional statements are handled by ***IF*** statements in Python.

if

```
if Expression1:#Expression is the conditions that you want to check for
BLOCK1 #It will be executed only if the above condition is true elif
Expression2:#It will check for other condition BLOCK2 else : #If none of
the above conditions satisfied then else statements will be executed
BLOCK3
```

```
if result ==1: print("I am one") elif result >1: print("I am greater than
one") else : print("I am less than one")
```

Don't Stop here 🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️🙅🏻‍♀️

Try to Code and Explore with Loops, Control, and Conditional Statements

✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅✅

Stay tuned !

FLOXUS EDUCATION,

Nurture the technical you