Python Programming

Session 9

Seyyed Ali Shohadaalhosseini

TOPICS

Python Fundamentals and Programming

- **Condition, loops**

Python Condition

What is the difference of = and == ?

Python conditions is:

1. Equal
2. Not Equal
3. Less than
4. Less than or equal to
5. Greater than
6. Greater than or equal to

Example

a = 20 and b = 22 what will be the answer of the following conditions:

Equals: a == b

Not equal: a != b

Less than: a < b

Less than or equal to: a <= b

Greater than: a > b

Greater than or equal to: a >= b

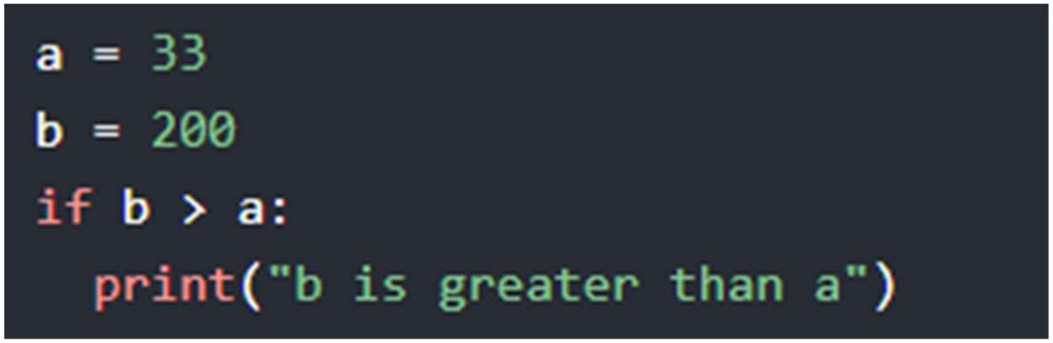
Python Condition

These conditions can be used in several ways, most commonly in “if statements” and loops.

We want to put a condition for a part of code to make that part only execute when the condition is true and to control the program we use “if statement” in python which its keyword is *if* .

If statements

In the following code we see that we choose the print line as a part of code and we decided to execute this line only when the variable b has a number greater than the a, and that’s how we write this:



```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```

Indentation

As you may notice, python relies on indentation(whitespace at the beginning of a line) to define scope in the code. Other programming languages often use curly-brackets for this purpose.

Else

Sometimes you put a condition in your code and you want to execute a part of your code when that condition is true, and also you want to know if the condition was not true another part of code executes only the time that condition wasn’t true, In this situation you can use the *else* keyword. Else is kind of condition that only runs when its if conditions was False. So it doesn’t have its own statements(condition).

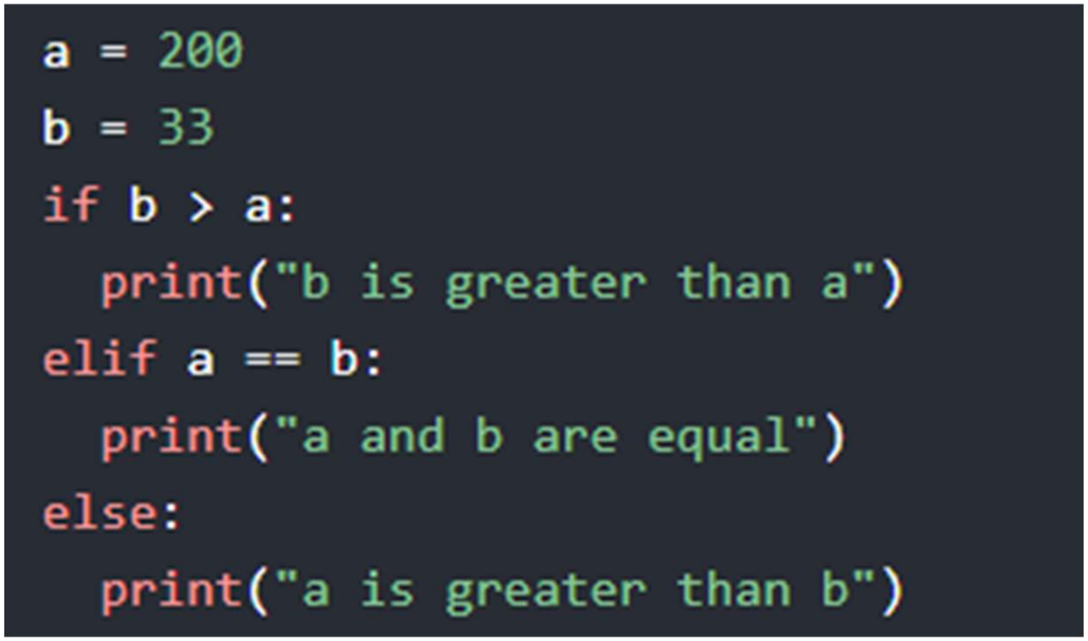
Elif

If you want your else to have its condition and only runs in the time when its conditions is true you can use the elif keyword and put its conditions right in front of it, exactly like what we had in the if statements.

Let’s see an example for that..

Example of if-elif-else

First of all if checks, if it’s not true then its elif checks for its conditions and if this elif is also not true then finally the else codes will run without any conditions.



```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

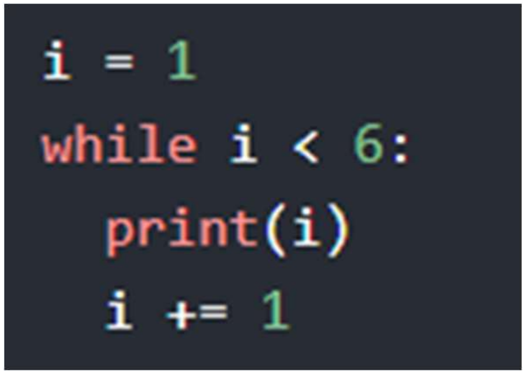
Loops

Python has two loop commands:

1. While loop
2. For loop

While loops

In the while loop we can execute a part of code that we have put it in the while, as long as while’s condition is true.

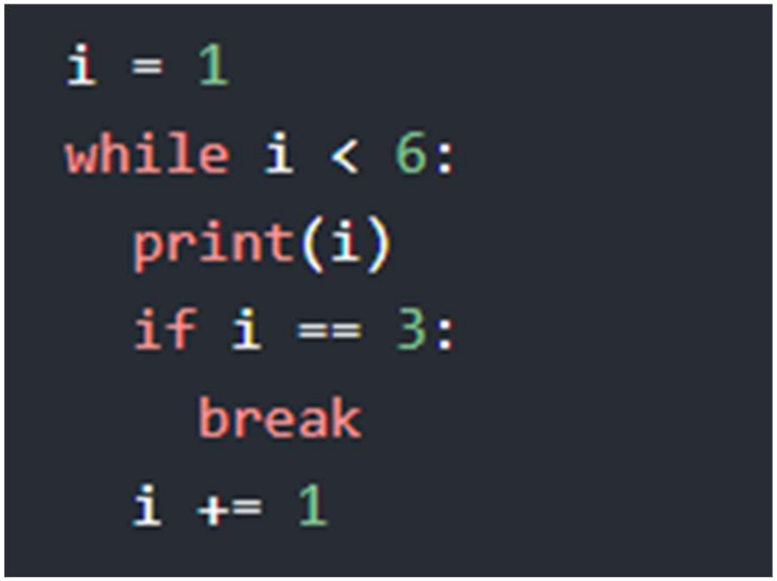


```
i = 1
while i < 6:
  print(i)
  i += 1
```

If you forget to increase the i, the loop will execute infinite and the program’s execution won’t stop.

The break statement

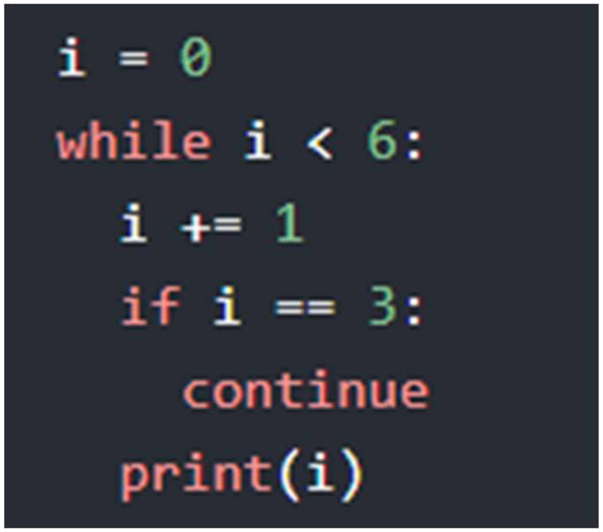
With the break statement we can stop the loop wherever we want in the loop even when the condition is true.



```
i = 1
while i < 6:
  print(i)
  if i == 3:
    break
  i += 1
```

The continue statements

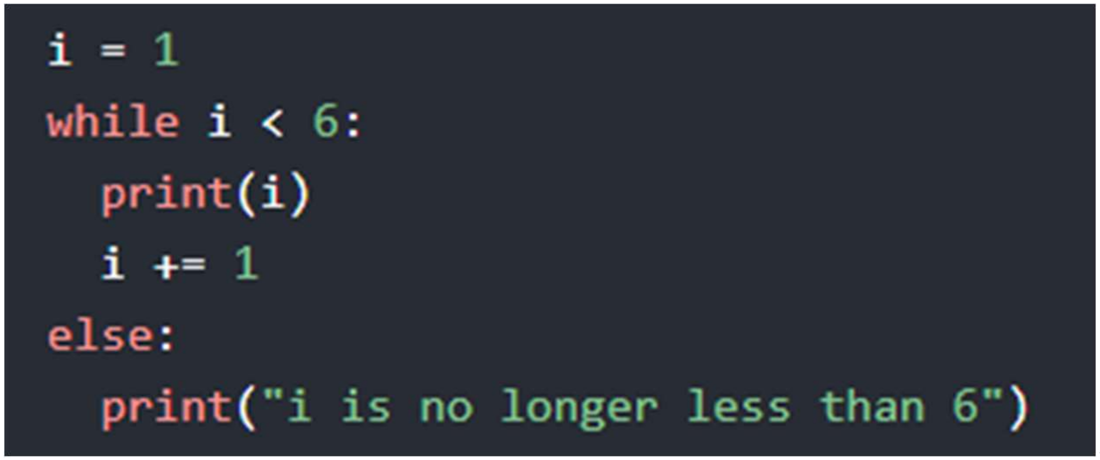
With the continue statement we can stop the current iteration, and start the next iteration immediately.



```
i = 0
while i < 6:
  i += 1
  if i == 3:
    continue
  print(i)
```

While loop’s else

While loop also have a else statement too. This statement only execute when the while’s conditions is not true anymore.

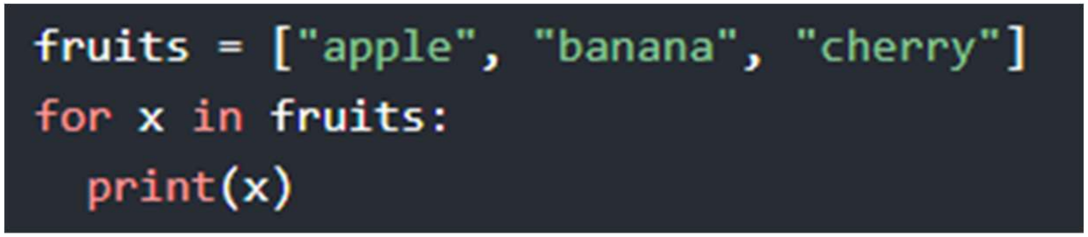


```
i = 1
while i < 6:
  print(i)
  i += 1
else:
  print("i is no longer less than 6")
```

For loops

For loop is used for iterating over a sequence.

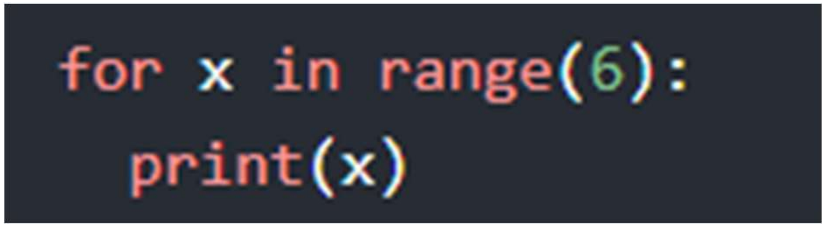
We can also iterate over variables that has multiple value too(like: iterating over a list, a tuple, a dictionary, a set, or a string).



```
fruits = ["apple", "banana", "cherry"]
for x in fruits:
  print(x)
```

The Range() method

The range function return a sequence of number over that number has been given to it. It starts from 0 by default and increments by 1, and ends at a specified number that has been send to it.



```
for x in range(6):
  print(x)
```