The Learning Hub

Control Flow Tools

Education and Training Solutions 2023

The Learning Hub

If Statements

Loop Statements

Range Function

Break, Continue & Pass Statements

The Learning Hub

If Statements

What is Python If Statement?

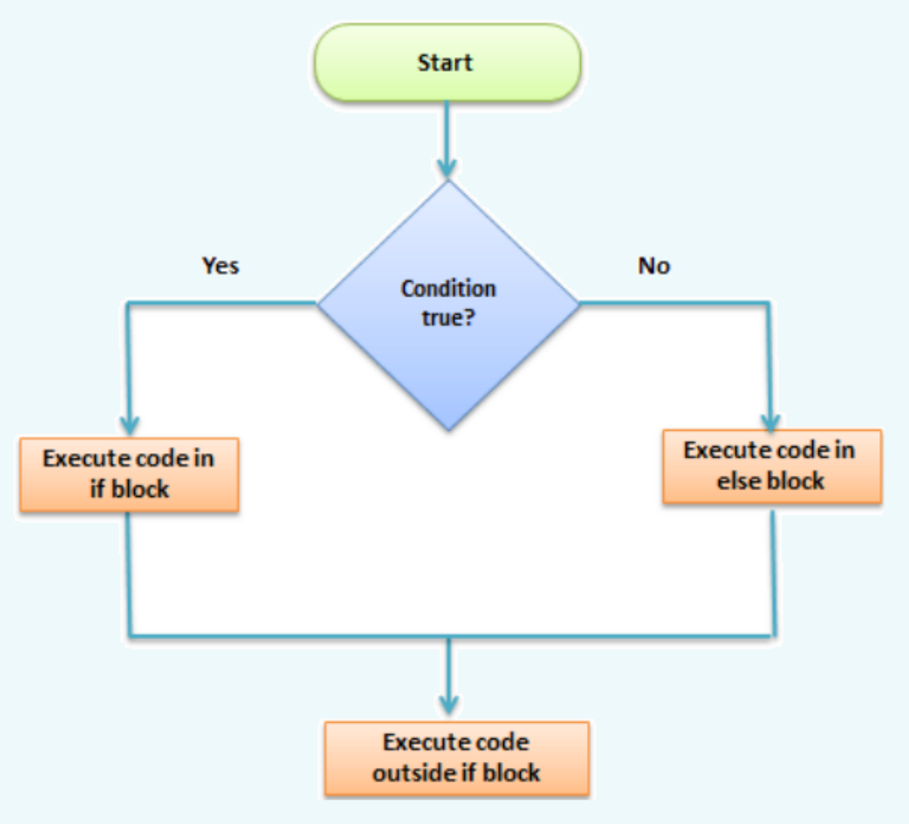
- Python if Statement: is used to make a decision. It contains a block of code in the body which will run only if the condition is true. If the condition is false then else statement will be executed.

- Python if Statement Syntax:

```
if expression
 Statement
else
 Statement
```

If Statement Flowchart

- Python if...else Flowchart



Logical Conditions

- Python supports of using the logical conditions:

- Equals: a == b
- Not Equals: a != b
- Less than: a < b
- Less than or equal to: a <= b
- Greater than: a > b
- Greater than or equal to: a >= b

Logical Conditions

- The logical conditions are most commonly used in If Statements and Loops Statements.

```
1  a = 50
2  b = 150
3
4  if b > a:
5        print("b is greater than a")
6
```

```
b is greater than a
```

Indentation

- Python uses an indentation as definition of the scope in the code. It is a whitespace at the beginning of each line of codes.

```
1  a = 50
2  b = 150
3
4  if b > a:
5  print("b is greater than a")
6
```

```
  Input In [9]
    print("b is greater than a")
    ^
IndentationError: expected an indented block
```

Elif

- The elif keyword is used in python to tell if the previous conditions were not true, then try this condition.

```
1  a = 88
2  b = 88
3  if b > a:
4        print("b is greater than a")
5  elif a == b:
6      print("a and b are equal")
```

```
a and b are equal
```

Else

- The else keyword is executed if all previous conditions were not true.

```
1  a = 150
2  b = 50
3  if b > a:
4      print("b is greater than a")
5  elif a == b:
6      print("a and b are equal")
7  else:
8      print("a is greater than b")
```

```
a is greater than b
```

Short Hand If ... Else

- You can have multiple else statements on the same line of code.

```
1  a = 0
2  b = 0
3  print("A") if a > b else print("=") if a == b else print("B")
```

```
=
```

And & Or keywords

- The and & or keywords are logical operators, used to combine conditional statements.

```
1  a = 100
2  b = 50
3  c = 150
4  if a > b and c > a:
5      print("Both conditions are True")
```

```
Both conditions are True
```

```
1  a = 100
2  b = 50
3  c = 150
4  if a > b or a > c:
5      print("At least one of the conditions is True")
```

```
At least one of the conditions is True
```

Nested If

- Nested if: You can have if statements inside if statements.

```
1  x = 20
2
3  if x > 10:
4      print("Above ten,")
5      if x > 20:
6          print("and also above 20!")
7      else:
8          print("but not above 20.")
```

```
Above ten,
but not above 20.
```

Pass Statement

- If statements can not be empty, if we have an empty if statement we can put the pass statement to prevent any errors.

```
1  a = 22
2  b = 140
3
4  if b > a:
5      pass
```

The Learning Hub

Loop Statements

Tahaluf Training Center 2022

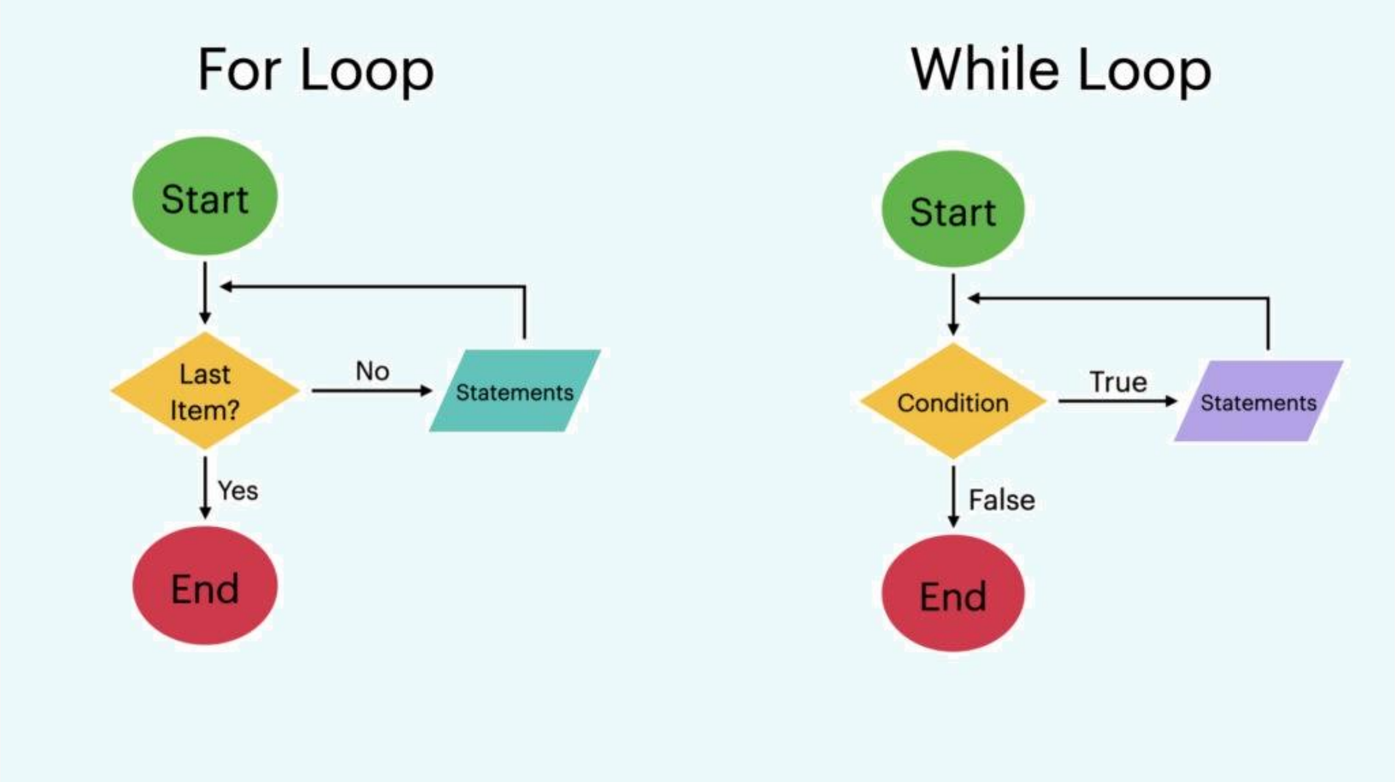
What is Loop?

- Loop: can execute a block of code by number of times until the end of condition. Python has just two direct loops commands (while, for loops).

- For Loop: is used to iterate over elements or items. It is used when you have a block of code which you want to repeat over number of time.

- While Loop: is used when you want to repeat a lines of codes. Instead of executing the block of code once, it executes the block multiple times until the end of condition.

For & while loops Flowchart



The while loop

- While loop can execute a number of statements while the condition is true.

```
1 i = 1
2 while i < 10:
3     print(i)
4     i += 1
```

```
1
2
3
4
5
6
7
8
9
```

The break statement

- Break statement can stop the while loop even if the loop condition is true.

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

```
1
2
3
```

The continue statement

- Continue statement can stop the current iteration and continue to the next.

```
1  i = 0
2  while i < 10:
3      i += 1
4      if i == 2:
5          continue
6      print(i)
```

```
1
3
4
5
6
7
8
9
10
```

The else statement in while loop

- else statement in while loop can run block of code once the condition is no longer is true.

```
1  i = 2
2  while i < 4:
3      print(i)
4      i += 1
5  else:
6      print("i is no longer less than 4")
```

```
2
3
i is no longer less than 4
```

The for loop

- For loop is used for iterating over a sequence (list, tuple, dictionary, set, and string).

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

```
apple
banana
cherry
```

Looping Through a String

- Strings are inerrable objects, so we can iterate on this sequence of characters.

```
1 for x in "banana":
2     print(x)
```

```
b
a
n
a
n
a
```

The break Statement

- We can stop the loop before looping through all the elements.

```
1  fruits = ["apple", "banana", "cherry"]
2  for x in fruits:
3      print(x)
4      if x == "banana":
5          break
```

```
apple
banana
```

The continue Statement

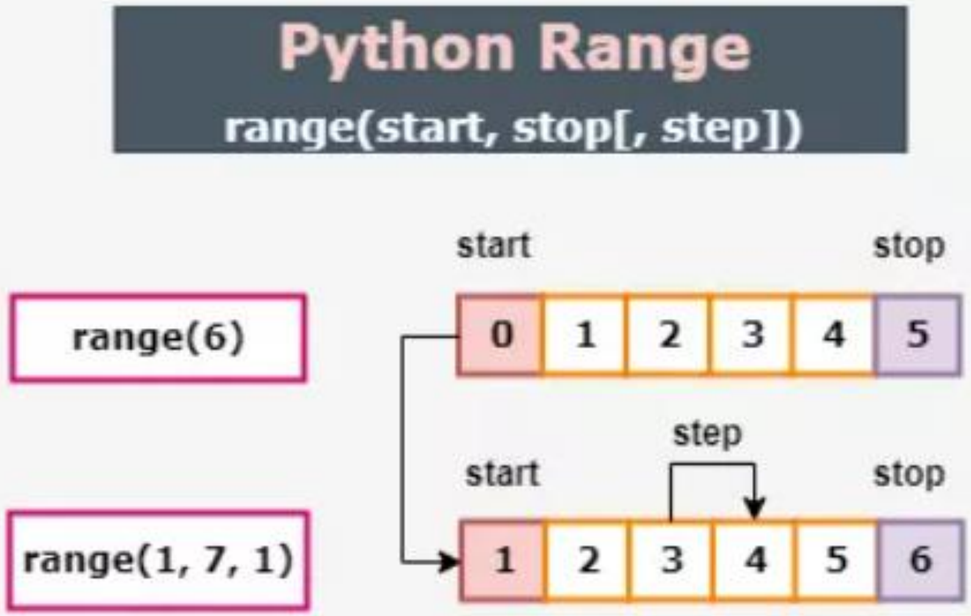
- We can stop the current iteration of the loop and continue for the next.

```
1  fruits = ["apple", "banana", "cherry"]
2  for x in fruits:
3      if x == "banana":
4          continue
5      print(x)
```

```
apple
cherry
```

The range Function

- Range function is used too loop through a block of codes with a specified number of times.



The range Function

- Range function default increment by 1, we can specify the increment by changing the third parameter.

```
1  for x in range(1, 15, 3):
2      print(x)
```

```
1
4
7
10
13
```

Else in For loop

- Else keyword in for loop specifies a block of codes to be executed when the loop is done.

```
1  for x in range(10):
2      print(x)
3  else:
4      print("Finally finished!")
```

```
0
1
2
3
4
5
6
7
8
9
Finally finished!
```

Nested Loops

- The inner loop will be executed one time for each iteration of the outer loop.

```
adj = ["red", "big", "tasty"]
fruits = ["apple", "banana", "cherry"]

for x in adj:
    for y in fruits:
        print(x, y)
```

```
red apple
red banana
red cherry
big apple
big banana
big cherry
tasty apple
tasty banana
tasty cherry
```

References

- Python Tutorial (w3schools.com)
- The Python Tutorial — Python 3.10.7 documentation
- Python Tutorial for Beginners: Learn Programming Basics [PDF] (guru99.com)
- https://www.codingem.com/flowchart-loop/
- https://www.techbeamers.com/python-range-function/

THANK YOU