PY0101EN-3-2-Loops

August 19, 2020

Loops in Python

Welcome! This notebook will teach you about the loops in the Python Programming Language. By the end of this lab, you'll know how to use the loop statements in Python, including for loop, and while loop.

```
<a href="https://cocl.us/topNotebooksPython101Coursera">
    <img src="https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClas
</a>
```

Table of Contents

```
<ul>
    <li>
        <a href="#loop">Loops</a>
        <ul>
            <li><a href="range">Range</a></li>
            <li><a href="for">What is <code>for</code> loop?</a></li>
            <li><a href="while">What is <code>while</code> loop?</a></li>
        </ul>
    </li>
    <li>
        <a href="#quiz">Quiz on Loops</a>
    </li>
</ul>
<p>
    Estimated time needed: <strong>20 min</strong>
</p>
```

Loops

Range

Sometimes, you might want to repeat a given operation many times. Repeated executions like this are performed by loops. We will look at two types of loops, for loops and while loops.

Before we discuss loops lets discuss the range object. It is helpful to think of the range object as an ordered list. For now, let's look at the simplest case. If we would like to generate a sequence that contains three elements ordered from 0 to 2 we simply use the following command:

```
[1]: # Use the range
```

```
range(3)
```

[1]: `range(0, 3)`

What is for loop?

The for loop enables you to execute a code block multiple times. For example, you would use this if you would like to print out every element in a list.
Let's try to use a for loop to print all the years presented in the list dates:

This can be done as follows:

[2]:
```
# For loop example

dates = [1982,1980,1973]
N = len(dates)

for i in range(N):
    print(dates[i])
```

```
1982
1980
1973
```

The code in the indent is executed N times, each time the value of i is increased by 1 for every execution. The statement executed is to print out the value in the list at index i as shown here:

In this example we can print out a sequence of numbers from 0 to 7:

[3]:
```
# Example of for loop

for i in range(0, 8):
    print(i)
```

```
0
1
2
3
4
5
6
7
```

In Python we can directly access the elements in the list as follows:

[4]:
```
# Exmaple of for loop, loop through list

for year in dates:
    print(year)
```

```
1982
1980
1973
```

For each iteration, the value of the variable years behaves like the value of dates[i] in the first example:

We can change the elements in a list:

```
[5]: # Use for loop to change the elements in list

     squares = ['red', 'yellow', 'green', 'purple', 'blue']

     for i in range(0, 5):
         print("Before square ", i, 'is',  squares[i])
         squares[i] = 'weight'
         print("After square ", i, 'is',  squares[i])
```

```
Before square  0 is red
After square  0 is weight
Before square  1 is yellow
After square  1 is weight
Before square  2 is green
After square  2 is weight
Before square  3 is purple
After square  3 is weight
Before square  4 is blue
After square  4 is weight
```

We can access the index and the elements of a list as follows:

```
[6]: # Loop through the list and iterate on both index and element value

     squares=['red', 'yellow', 'green', 'purple', 'blue']

     for i, square in enumerate(squares):
         print(i, square)
```

```
0 red
1 yellow
2 green
3 purple
4 blue
```

What is while loop?

As you can see, the for loop is used for a controlled flow of repetition. However, what if we don't know when we want to stop the loop? What if we want to keep executing a code block until a certain condition is met? The while loop exists as a tool for repeated execution based on a condition. The code block will keep being executed until the given logical condition returns a **False** boolean value.

Let's say we would like to iterate through list dates and stop at the year 1973, then print out the number of iterations. This can be done with the following block of code:

```
[7]:  # While Loop Example

      dates = [1982, 1980, 1973, 2000]

      i = 0
      year = 0

      while(year != 1973):
          year = dates[i]
          i = i + 1
          print(year)

      print("It took ", i ,"repetitions to get out of loop.")
```

```
1982
1980
1973
It took  3 repetitions to get out of loop.
```

A while loop iterates merely until the condition in the argument is not met, as shown in the following figure:

Quiz on Loops

Write a for loop the prints out all the element between -5 and 5 using the range function.

```
[9]:  # Write your code below and press Shift+Enter to execute
      for i in range(-5,6):
          print(i)
```

```
-5
-4
-3
-2
-1
0
1
2
3
4
5
```

Double-click **here** for the solution.

Print the elements of the following list: Genres=['rock', 'R&B', 'Soundtrack', 'R&B', 'soul', 'pop'] Make sure you follow Python conventions.

```
[12]:  # Write your code below and press Shift+Enter to execute
       Genres=[ 'rock', 'R&B', 'Soundtrack', 'R&B', 'soul', 'pop']
       for elements in Genres:
           print(elements)
```

```
rock
R&B
Soundtrack
R&B
soul
pop
```

Double-click **here** for the solution.

Write a for loop that prints out the following list: squares=['red', 'yellow', 'green', 'purple', 'blue']

```
[14]:  # Write your code below and press Shift+Enter to execute
       squares=['red', 'yellow', 'green', 'purple', 'blue']

       for elements in squares:
           print(elements)
```

```
red
yellow
green
purple
blue
```

Double-click **here** for the solution.

Write a while loop to display the values of the Rating of an album playlist stored in the list PlayListRatings. If the score is less than 6, exit the loop. The list PlayListRatings is given by: PlayListRatings = [10, 9.5, 10, 8, 7.5, 5, 10, 10]

```
[ ]:  # Write your code below and press Shift+Enter to execute
      PlayListRatings = [10, 9.5, 10, 8, 7.5, 5, 10, 10]
      i=0
      while (PlayListRatings[i]>6):
          print(PlayListRatings[i])
          i=i+1
```

Double-click **here** for the solution.

Write a while loop to copy the strings 'orange' of the list squares to the list new_squares. Stop and exit the loop if the value on the list is not 'orange':

```
[ ]:  # Write your code below and press Shift+Enter to execute

      squares = ['orange', 'orange', 'purple', 'blue ', 'orange']
      new_squares = []
```

```
i=0
while (squares[i]=='orange')and(i<=len(squares):
    new_squares.append('orange')
    i=i+1
print(new_squares)
```

Double-click **here** for the solution.

The last exercise!

Congratulations, you have completed your first lesson and hands-on lab in Python. However, there is one more thing you need to do. The Data Science community encourages sharing work. The best way to share and showcase your work is to share it on GitHub. By sharing your notebook on GitHub you are not only building your reputation with fellow data scientists, but you can also show it off when applying for a job. Even though this was your first piece of work, it is never too early to start building good habits. So, please read and follow this article to learn how to share your work.

Get IBM Watson Studio free of charge!

```
<p><a href="https://cocl.us/bottemNotebooksPython101Coursera"><img src="https://s3-api.us-geo.
```

About the Authors:

Joseph Santarcangelo is a Data Scientist at IBM, and holds a PhD in Electrical Engineering. His research focused on using Machine Learning, Signal Processing, and Computer Vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Other contributors: Mavis Zhou, James Reeve