Python Programming

We have been through the basics in Python, such as variables, using some basic built-in functions, basic plotting, etc.

You may come far only using these thins, but to create real applications, you need to know about and use features like:

• If ... Else

• For Loops

• While Loops

• Arrays ...

If you are familiar with one or more other programming language, these features should be familiar and known to you. All programming languages has these features built-in, but the syntax is slightly different from one language to another.

5.1 If ... Else

An ”if statement” is written by using the if keyword.

Here are some Examples how you use a If sentences in Python:

Example 5.1.1. Using For Loops in Python

```
a = 5
b = 8

ifa>b:
    print("a is greater than b")

if b> a:
    print("b is greater than a")

if a == b:
    print("a is equal to b")
```

Listing 5.1: Using Arrays in Python

Try to change the values for a and b.

Using If - Else:

```
a = 5
b = 8

if a>b:
    print("a i  greate  than b")
else:
    print("b s  r       than a or  a and b are  equal")
```

Listing 5.2: Using Arrays in Python

```
          i  greate

          s  r
```

Using Elif :

```
a = 5
b = 8

if a>b:
    print("a i  greater than b")
elif b > a:
    print("b s  greater than a")
   elif a==b:
    print("a i  equal to b")
```

Listing 5.3: Using Arrays in Python

```
          s

          i
```

Note! Python uses ”elif” not ”elseif” like many other programming languages do.
```
          s
```

[End of Example]

5.2 Arrays

An array is a special variable, which can hold more than one value at a time.

Here are some Examples how you can create and use Arrays in Python:

Example 5.2.1. Using For Loops in Python

```
data = [1.6, 3.4, 5.5, 9.4]

N = len(data)

print (N)

print(data[2])

data[2] = 7.3

print(data[2])


for x in data:
print(x)
```

```
16
17
18 data . append ( 11 . 4 )
19
20
21 N           =
22
23 len(data)
24
25 print (N)
26 for x in data:
27 print(x)
```

Listing 5.4: Using Arrays in Python

You define an array like this:

```
1 data = [1.6,   3.4,  5.5,  9.4]
```

You can also use text like this:

```
1 carlist = ["Volvo",   "Tesla",   "Ford"]
```

You can use Arrays in Loops like this:

```
1 for x in data:
2 print(x)
```

You can return the number of elements in the array like this:

```
1 N = len(data)
```

You can get a specific value inside the array like this:

```
1 index = 2
2 x = cars[index]
```

You can use the append() method to add an element to an array:

```
1 data . append ( 11 . 4 )
```

[End of Example]

You have many built in methods you can use in combination with arrays, like sort(), clear(), copy(), count(), insert(), remove(), etc.

You should look test all these methods.

5.3 ForLoops

A For loop is used for iterating over a sequence. I guess all your programs will use one or more For loops. So if you have not used For loops before, make sure to learn it now.

Below you see a basic example how you can use a For loop in Python:

```
for i in range(1,   10):
print(i)
```

The For loop is probably one of the most useful feature in Python (or in any kind of programming language). Below you will see different examples how you can use a For loop in Python.

Example 5.3.1. Using For Loops in Python

```
data = [1.6, 3.4,    5.5,  9.4]

for x in data:
print(x)


carlist = ["Volvo",   "Tesla",  "Ford"]

for car in carlist:
print(car)
```

Listing 5.5: Using For Loops in Python

The range() function is handy yo use in For Loops:

```
N=10

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

The range() function returns a sequence of numbers, starting from 0 by default, and increments by 1 (by default), and ends at a specified number.

You can also use the range() function like this:

```
start = 4
stop= 12 #but not including

for x in range(start, stop):
print(x)
```

Finally, you can also use the range() function like this:

```
start = 4
stop = 12 #but not including
step = 2

for x in range(start, stop, step):
print(x)
```

You should try all these examples in order to learn the basic structure of a For lo
op.

[End of Example]

Example 5.3.2. Using For Loops for Summation of Data

You typically want to use a For loop for find the sum of a given data set.

```
data = [1, 5, 6, 3, 12, 3]

sum = 0

#Find the Sum of all the numbers for x in
data:
sum = sum + x

print(sum)

#Find the Mean or Average of all
                                    the numbers

N = len(data)

mean = sum/N

print (mean)
```

This gives the following results:

```
30
5.0
```

[End of Example]

Example 5.3.3. Implementing Fibonacci Numbers Using a For Loop in Python

Fibonacci numbers are used in the analysis of financial markets, in strategies such as Fibonacci retracement, and are used in computer algorithms such as the
Fibonacci search technique and the Fibonacci heap data structure.
They also appear in biological settings, such as branching in trees, arrangement of leaves on a stem, the fruitlets of a pineapple, the flowering of artichoke, an uncurling fern and the arrangement of a pine cone.

In mathematics, Fibonacci numbers are the numbers in the following sequence:

0, 1, 1, 2 ,3, 5, 8, 13, 21, 34, 55, 89, 144, . . .

By definition, the first two Fibonacci numbers are 0 and 1, and each subsequent number is the sum of the previous two.

Some sources omit the initial 0, instead beginning the sequence with two 1s.

In mathematical terms, the sequence Fn of Fibonacci numbers is defined by the recurrence relation

$$f_n = f_{n-1} + f_{n-2} \tag{5.1}$$

with seed values:

f0=0,f1=1

We will write a Python script that calculates the N first Fibonacci numbers. The Python Script becomes like this:

```
N=10

fib1 = 0
fib2 = 1
print(fib1)
print(fib2)
for k in range(N
-2):
    fib next = fib2 +fib1
fib1 = fib2
fib2 = fib next
print(fib next)
```

Listing 5.6: Fibonacci Numbers Using a For Loop in Python

Alternative solution:

```
N=10

fib = [0,  1]


for k in range(N
-2):
    fib next = fib[k+1] +fib[k]
fib.append(fibnext)
print(fib)
```

Listing 5.7: Fibonacci Numbers Using a For Loop in Python - Alt2

Another alternative solution:

```
N=10

fib = []

for k in range(N):
fib .append(0)

fib[0] = 0
fib[1] = 1

```

```
for k in range(N
-2):
    fib [k+2] = fib [k+1] +fib [k]


print(fib)
```

Listing 5.8: Fibonacci Numbers Using a For Loop in Python - Alt3

Another alternative solution:

```
import numpy as np


N=10

fib = np.zeros(N)

fib[0] = 0
fib[1] = 1
for k in range(N
-2):
    fib [k+2] = fib [k+1] +fib [k]


print(fib)
```

Listing 5.9: Fibonacci Numbers Using a For Loop in Python - Alt4

[End of Example]

5.3.1 NestedForLoops

In Python and other programming languages you can use one loop inside another loop.

Syntax for nested For loops in Python:

```
for iterating_var in sequence :
    for iterating_var in sequence:
statements(s)
statements(s)
```

Simple example:

```
for i in range(1, 10):
    for k in range(1, 10):
    print(i, k)
```

Exercise 5.3.1. Prime Numbers

The first 25 prime numbers (all the prime numbers less than 100) are:
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

By definition a prime number has both 1 and itself as a divisor. If it has any other divisor, it cannot be prime.

A natural number (1, 2, 3, 4, 5, 6, etc.) is called a prime number (or a prime) if it is greater than 1 and cannot be written as a product of two natural numbers that are both smaller than it.

Create a Python Script where you find all prime numbers between 1 and 200.

Tip! I guess this can be done in many different ways, but one way is to use 2 nested For Loops.

[End of Exercise]

5.4 WhileLoops

The while loop repeats a group of statements an indefinite number of times under control of a logical condition.

Example 5.4.1. Using While Loops in Python

```
m = 8

while m > 2:
    print (m)
    m = m - 1
```

Listing 5.10: Using While Loops in Python

[End of Example]

5.5 Exercises

Below you find different self-paced Exercises that you should go through and solve on your own. The only way to learn Python is to do lots of Exercises!

Exercise 5.5.1. Plot of Dynamic System

Given the autonomous system:

$$\dot{x} = ax \tag{5.2}$$

Where:

$a = -\frac{1}{T}$

where T is the time constant.

The solution for the differential equation is:

$$x(t) = x_0 e^{-\frac{t}{T}} \qquad (5.3)$$

Set T=5 and the initial condition x(0)=1.

Create a Script in Python (.py file) where you plot the solution x(t) in the time interval:

$$0 \leq t \leq 25$$

Add Grid, and proper Title and Axis Labels to the plot.

[End of Exercise]