

Introduction to Python



Lucas MORLET
BSB : M1

WHO AM I ?

Studies

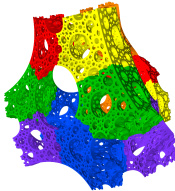
- 2011-2016 : Master of Images and Artificial Intelligence
- 2019 : Ph.D. of Computer Sciences at UBE
(Computer Graphics - Geometric Modeling)

Teacher Career

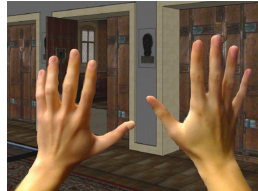
- 2019-2020 : Junior Lecturer at UBE
- since 2020 : Lecturer at ESEO Dijon
- since 2020 : In charge of Smart City Major
- since 2021 : Independent contractor at ESTP Dijon
- since 2023 : Independent contractor at BSB Dijon

INTRODUCTION

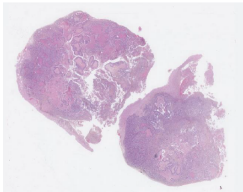
Geometric Modeling



Mixed Realities



Medical Imaging



Smart City



COURSE PROGRESS


Introduction to Python :

- 24 sept. : variables, conditions, loops, and functions
- 08 oct. : 1D and 2D lists
- 15 oct. : strings and dictionaries













Data Science :



- 22 oct. : dataframes and data preparation
- 12 nov. : basics of statistics and data viz
- 19 nov. : correlation, curve fitting and features selection
- 26 nov. : supervised clustering
- 03 dec. : unsupervised clustering
- 10 dec. : project final rush

WHAT IS PYTHON ?



Python Advantages and Disadvantages

| Advantages  | Disadvantages  |
|--|--|
|  Improved Productivity |  Slow Speed |
|  Interpreted Language |  Not Memory Efficient |
|  Dynamically Typed |  Weak in Mobile Computing |
|  Free and Open Source |  Database Access |
|  Vast Libraries Support |  Runtime Errors |



INTRODUCTION - PYTHON

YOU SAID "LIBRAIRIES" ?

Top 10 Python Libraries



Pandas

Data analysis and manipulation



NumPy

Mathematical functions



Matplotlib

Data visualisations



SeaBorn

Data visualisations



Tensorflow

Machine Learning



Keras

Deep Learning



SciPy

Scientific computing



PyTorch

Machine Learning



Scrapy

Web crawling

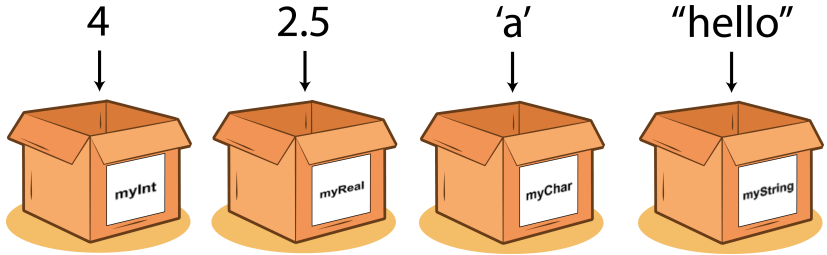


SQLModel

Interact with SQL databases

 DATA RUNDOWN

VARIABLES



ASSIGN A VALUE TO A VARIABLE

Create a number variable

```
number = 8.4
```

Create a character variable

```
char = 'A'
```

Create a text variable

```
string = "test"
```

Print the content of the variables

```
print ( number, char, string )
```

Results

8.4 A test

INTEGERS (INT)

Assign values

n = 3

m = 2

Print the sum

print (n + m)

Print the type of the variable

print (type(n))

Results

5

< class 'int' >

DECIMAL NUMBER (FLOAT)

Create a float

```
x = 3.14
```

Print it and its type

```
print ( x, type(x) )
```

Mix integers to get a float

```
n = 3
```

```
m = 2
```

```
print ( n/m, type(n/m) )
```

Results

```
3.14 < class 'float' >
```

```
1.5 < class 'float' >
```

BOOLEANS (BOOL)

Create two booleans

a = True

b = False

Print a bool and its type

print (a, type(a))

Some operations on booleans

print (not a, a and b, a or b)

Result

True <class 'bool'>

False False True

STRING (STR)

Create two strings

```
s = "Hello"
```

```
t = "World !"
```

Print it

```
print ( s, t )
```

```
print ( type(s) )
```

Result

Hello World !

<class 'str'>

DYNAMIC TYPE

Create an integer

```
x = 4
```

```
print ( x, end = " " )
```

Convert it to float

```
x = 8.6
```

```
print ( x, end = " " )
```

Finally, its a string !

```
x = "I'm a string"
```

```
print ( x )
```

Result

4 8.6 I'm string

CONDITIONAL STRUCTURES



STANDARD STRUCTURE

```
x = 2
if ( x == 3 ) :
    print ( "That's true" )
else :
    print ( "That's false" )
```

Results

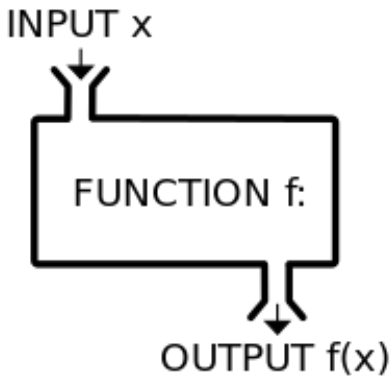
That's false

MULTIPLE ENDS STRUCTURE

```
x = 2
if ( x == 3 ) :
    print ( "That's true" )
elif ( x == 2 ) :
    print ( "That's false then true" )
else :
    print ( "That's false" )
```

Results

That's false then true



SOME BASIC FUNCTION

Function definition

```
def my_function ( ) :  
    print ( "I'm inside the function" )
```

Call this function

```
print ( "I will run some function" )  
my_function ( )
```

Results

```
I will run some function  
I'm inside the function
```

PASS A PARAMETER TO A FUNCTION

Define the function

```
def two_times ( n ) :  
    print ( "2 x ", n , "=", 2*n )
```

Test it with different parameters

```
two_times ( 3 )  
two_times ( 4 )
```

Results

2 x 3 = 6

2 x 4 = 8

PASS SEVERAL PARAMETERS TO A FUNCTION

Define the function

```
def sum ( a, b ) :  
    print ( a, "+", b, "=", a+b )
```

Test it with different parameters

```
sum ( 2, 3 )  
sum ( 1.4, 2.1 )
```

Results

$2 + 3 = 5$

$1.4 + 2.1 = 3.5$

RETURN

Define the function

```
def square ( x ) :  
    return x*x
```

Test it with different parameters

```
y = square ( 2 )  
print ( y )  
y = square ( 2.5 )  
print ( y )
```

Results

4

6.25

VARIABLE SCOPE

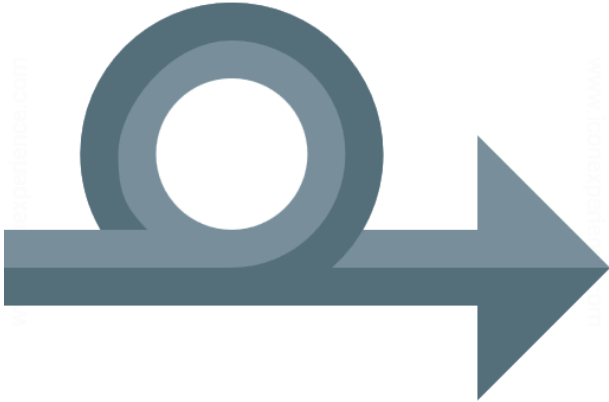
```
def some_function ( x ) :  
    x = 2 * x  
    print ( "During the function : ", x )
```

```
x = 3  
print ( Before the function, x )  
some_function ( x )  
print ( After the function, x )
```

Results

Avant la fonction : 3
Pendant la fonction : 6
Après la fonction : 3

LOOPS



www.iconexperience.com

FOR-LOOP THROUGH A SET

```
# For a specif set of values  
for i in [ 2, 3, 5, 7, 11 ] :  
    print ( i, end = " " )
```

Results

2 3 5 7 11

FOR-LOOP THROUGH A RANGE

From 0 to n-1

```
for i in range ( 6 ) :  
    print ( i, end = " " )  
( " " )
```

Results

0 1 2 3 4 5

From m to n-1

```
for i in range ( 2, 8 ) :  
    print ( i, end = " " )  
( " " )
```

Results

2 3 4 5 6 7

With a specific step between numbers

```
for i in range ( 0, 8, 2 ) :  
    print ( i, end = " " )  
( " " )
```

Results

0 2 4 6

WHILE LOOP

```
n = 18
while ( n > 1 ) :
    print ( n, end = " " )
    n = n / 2
print ( " " )
```

Results

18, 9.0, 4.5, 2.25, 1.125

NESTED LOOPS

```
for i in range ( 1, 3 ) :  
    for j in [ 'a', 'b', 'c' ] :  
        print ( i, j, end = " , " )  
print ( " " )
```

Results

1 a, 1 b, 1 c, 2 a, 2 b, 2 c

LINKED LOOPS

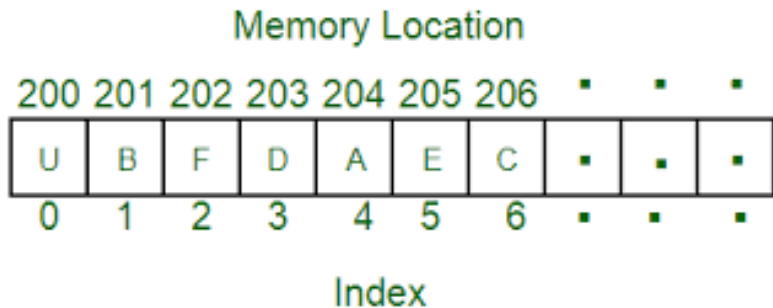
```
for i in range ( 1, 5 ) :  
    for j in range ( 2 * i ) :  
        print ( " ", end = "" )  
    print ( " ", end = "" )
```

Results

..

(2 dots then 4 dots then 6 dots and to finish 8 dots)

ARRAY



WHAT IS IT ?

- An array is a data structure
- It is composed of a fixed number of same type elements
- Each element contains a value
- Each element should be accessed by its index

Be careful !

In Python, there is no **array**, only **list**

Lists are similar to array, without the fixed number constraint

In a first part, you will use Python's list as array

Later, you will learn how to exploit list functionalities

CREATE AN ARRAY

Define an array composed of ten zeros

```
tab = [0] * 10
```

Display the array infos

```
print ( "The array is :", tab )
```

```
print ( "Its length is :", len(tab) )
```

Results

The array is : [0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

Its length is : 10

READ THE VALUE OF AN ELEMENT

```
# Define the array [ a, b, c, d, e ]  
tab = [ 'a', 'b', 'c', 'd', 'e' ]  
# Display the array infos  
print ( "The array is :", tab )  
print ( "The element number 1 is :", tab[1] )
```

Results

The array is : ['a', 'b', 'c', 'd', 'e']
The element number 1 is : b

STORE A VALUE INSIDE AN ELEMENT

Define an array composed of five zeros

```
tab = [0] * 5
```

```
print ( "Before modification :", tab )
```

Store a 2 inside element number 2

```
tab[2] = 2
```

```
print ( "After modification :", tab )
```

Results

Before modification : [0, 0, 0, 0, 0]

After modification : [0, 0, 2, 0, 0]

MODIFY ELEMENTS OF AN ARRAY

Define an array

```
tab = [ 2, 3, 5, 7, 11 ]
```

```
print ( "Original array :", tab )
```

Multiply element number 3 by 10

```
tab[3] = 10 * tab[3]
```

```
print ( "After multiplication :", tab )
```

Sum two elements of the array

```
tab[2] = tab[1] + tab[2]
```

```
print ( "After sum :", tab )
```

Results

Original array : [2, 3, 5, 7, 11]

After multiplication : [2, 3, 5, 70, 11]

After sum : [2, 3, 8, 70, 11]

ITERATE THROUGH AN ARRAY

Define an array composed of ten zeros

```
tab = [0] * 10
```

Define every element by iterating through the array

```
for i in range ( len ( tab ) ) :
```

```
    tab [ i ] = i
```

```
print ( "After iteration :" , tab )
```

Results

After iteration : [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

MODIFY AN ARRAY BY ITERATION

Previous slide code

```
tab = [0] * 10
```

```
for i in range ( len ( tab ) ) :  
    tab [ i ] = i
```

Square every element of the array

```
for i in range ( len ( tab ) ) :  
    tab [ i ] = tab [ i ] * tab [ i ]
```

```
print ( "Finally :" , tab )
```

Results

```
Finally : [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```