INTRODUCTION À PYTHON

2 PYTHON, C'EST QUOI?

- Première version sortie en 1991
- Langage interprété
  - Simple à comprendre
  - Modulable
  - Applications multiples (POO, data science, backend, …)
- Langage full stack

3 ENVIRONNEMENT DE TRAVAIL

- https://www.python.org/downloads/ - Python3
- Powershell => indique si python3 est déjà sur la machine ou non

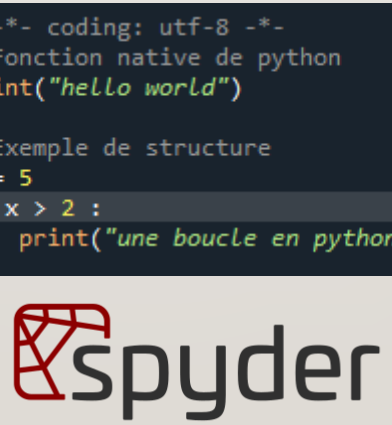
4 EXÉCUTION D'UN SCRIPT PYTHON

```python
#fonction native en python
print('Hello world')

#exemple de structure
x = 5
if x > 3 :
    print('x est supérieur à 3!')

#On lance le déboggage
```

```
\lib\python\debugpy\lau
Hello world
x est supérieur à 3!
```

```
In [2]: runfile('C:/Users/
hello world
une boucle en python
```

```python
# -*- coding: utf-8 -*-
# Fonction native de python
print("hello world")

# Exemple de structure
x = 5
if x > 2 :
    print("une boucle en python")
```



 # Une fonction native de *python*

Entrée [1]:
```python
print("Hello World")
```
```
Hello World
```

Un exemple de structure

Entrée [2]:
```python
x = 5
if x > 3 :
    print("le nombre est supérieur")
```
```
le nombre est supérieur
```

5 LES VARIABLES EN PYTHON

- Pour assigner une variable, on utilise `=`
- Quelques règles pour nommer les variables
  - Commencer par lettre / underscore
  - Contenir que des caractères alphanumériques (**PAS D'ESPACES OU D'ACCENT**!!)
  - Attention aux mots réservés https://fr.wikibooks.org/wiki/Programmation_Python/Tableau_des_mots_r%C3%A9serv%C3%A9s
- Pour afficher le contenu, on utilise la fonction native print( )

6 LES VARIABLES EN PYTHON

- Type numérique
  - Type int
  - Type float
  - Type complex
- Type string
- Type booléen
- Fonction native de type

```
x = 5
y = "coucou"
z = True

type(x)
type(y)
type(z)
```

```
In [12]: type(x)
Out[12]: int

In [13]: type(y)
Out[13]: str

In [14]: type(z)
Out[14]: bool
```

Nar	Type	Size	Value
x	int	1	5
y	str	6	coucou
z	bool	1	True

7 LES LISTES

- Permet de stocker plusieurs variables de différents types
- ~ array en JS et PHP

```
list = {"BeCode", 2022, True}
print(list)
```

8 LES TUPLES

- Immuable après sa création contrairement aux listes
- Si on veut que nos données ne soient pas modifiées dans un programme, on partira vers des tuples au lieu des listes
- Déballage de séquences

```python
tuple = ("becode", 2022, True)
nom, annee, code = tuple

print(nom)
print(annee)
print(code)
```

```
becode
2022
True
```

9 LES DICTIONNAIRES

```
# var = {}
# var = dict()
```

- ~ array associatif en PHP
- Fonctionne avec un système de clé et de valeur

```
langage = {'front':'HTML', 'back':'PHP', 'framework':'Laravel'}
langage2 = dict(front='HTML', back='PHP', framework='Laravel')

print(langage['back'])
```

10 RÉCAPITULATIF SUR LES LISTES, LES TUPLES ET LES DICTIONNAIRES

- Les listes sont des collections d'éléments modifiables pouvant contenir plusieurs fois la même valeur. De plus, les listes sont ordonnées.

- Les tuples sont des collections d'éléments immuables pouvant contenir plusieurs fois la même valeur. De plus, les tuples sont ordonnés.

- Les dictionnaires sont des collections d'éléments indexés avec des clés. De plus, un dictionnaire est modifiable mais n'accepte pas plusieurs fois le même élément.

11 LES BOUCLES

ATTENTION A L INDENTATION

- If…else
- If…elif…else
- Opérateur d'appartenance

```
if x < 100:
    print('La variable est inférieure à 100')
elif x == 100:
    print('La variable est égale à 100')
else :
    print('La variable est supérieure à 100')
```

```
langage_programmation = ["python", "R", "SQL", "PHP"]

if "HTML" in langage_programmation:
    print("HTML est présent")
else:
    print("HTML n'est pas présent dans la liste")
```

- Boucle while
- Boucle for

```
x = 0
while x < 5:
    print(x)
    x+=1
```

```
langage_programmation = ["python", "R", "SQL", "PHP"]
for i in langage_programmation:
    print(i)
```

```
for n in range(10):
    if n % 2 == 1:
        continue
    else:
        print(n)
```

12 LES FONCTIONS

- DRY
- Mot clé *def*

```
def somme(x,y):
    print(x+y)

somme(4,5)
```

```
def somme(x,y):
    print(x+y)

s=[1,2]
somme(*s)
```

- Principe du passage d'un nombre arbitraire d'arguments
  - *args : permet d'indiquer que notre fonction peut accepter un nombre variable d'arguments (TUPLE)
  - **kwargs : permet d'indiquer que notre fonction peut accepter un nombre variable d'arguments (DICTIONNAIRE)

```
def somme(*args):
    s = 0
    for n in args:
        s += n
        print("La somme vaut : ", s)

somme(15, 19, 145, 1, 42)
```

```
def presentation(**kwargs):
    for i, j in kwargs.items():
        print(i,j)

presentation(Prénom='Loïc', age=29, ville='La Louvière')
```

13 LA PROGRAMMATION ORIENTÉE OBJET

- __init()__ = constructeur
- Méthodes getter et setter
- Self ~ this en PHP

```python
class Utilisateur:
    def __init__(self, prenom, age):
        self.prenom = prenom
        self.age = age

    def getName(self):
        print("Salut, je suis", self.prenom)

class Presentation(Utilisateur):
    def getPresentation(self):
        print("Salut, je suis", self.prenom, "et j'ai", self.age)

moi = Presentation('Loïc', 29)
moi.getPresentation()
```

```python
class Utilisateur:
    def __init__(self, prenom, age):
        self.prenom = prenom
        self.age = age

    def getName(self):
        print("Salut, je suis", self.prenom)

moi = Utilisateur('Loïc', 29)
moi.getName()
```

14 LES MODULES EN PYTHON

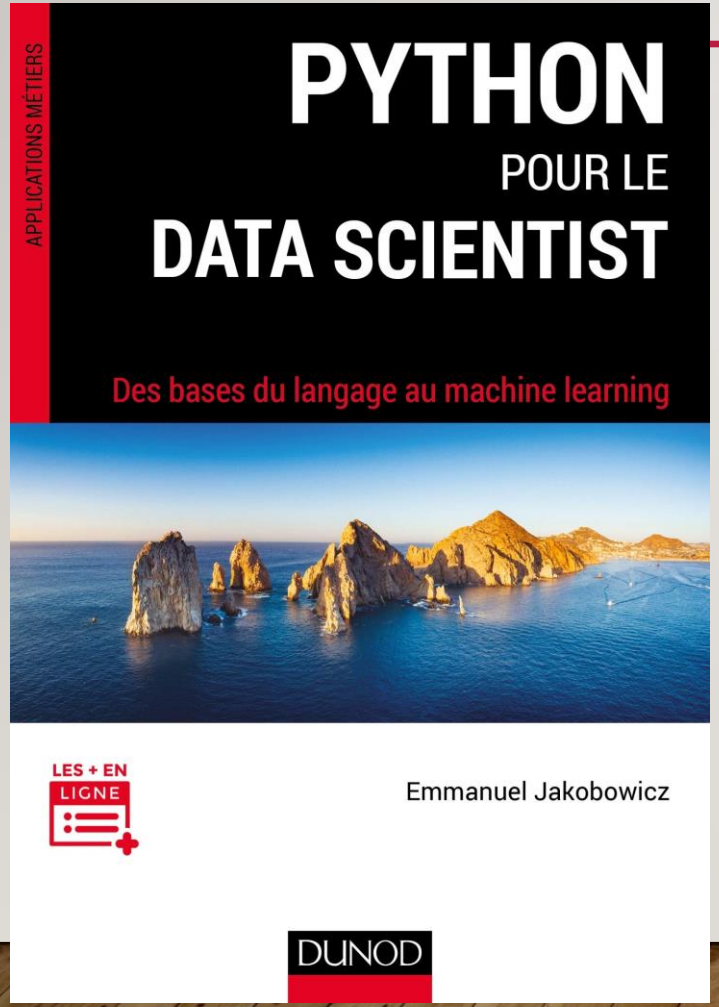
- Un module c'est un fichier de code python
- Module de base implémenté dans python : re, math, json, os, datetime, …

- Les modules peuvent se retrouver dans des packages
  - Exemple : le module pyplot dans le package matplotlib

```
import matplotlib.pyplot as plt
x = [1, 2, 2, 3, 4, 4, 4, 4, 4, 5, 5]
plt.hist(x, range = (0, 5), bins = 5, color = 'red', edgecolor='white')
plt.xlabel('valeurs')
plt.ylabel('nombres')
plt.title('Exemple d\' histogramme simple')
```

15 QUELQUES RESSOURCES…

Script de la veille : https://github.com/CalcagnoLoic/veille-becode/blob/main/Introduction%20%C3%A0%20python.py



- https://openclassrooms.com/fr/courses/6951236-mettez-en-place-votre-environnement-python
- https://openclassrooms.com/fr/courses/7168871-apprenez-les-bases-du-langage-python
- https://openclassrooms.com/fr/courses/4302126-decouvrez-la-programmation-orientee-objet-avec-python
- https://openclassrooms.com/fr/courses/7150616-apprenez-la-programmation-orientee-objet-avec-python
- https://openclassrooms.com/fr/courses/7172076-debutez-avec-le-framework-django
- https://openclassrooms.com/fr/courses/4425066-concevez-un-site-avec-flask
- https://www.pierre-giraud.com/python-apprendre-programmer-cours/

MERCI POUR VOTRE ATTENTION ☺