

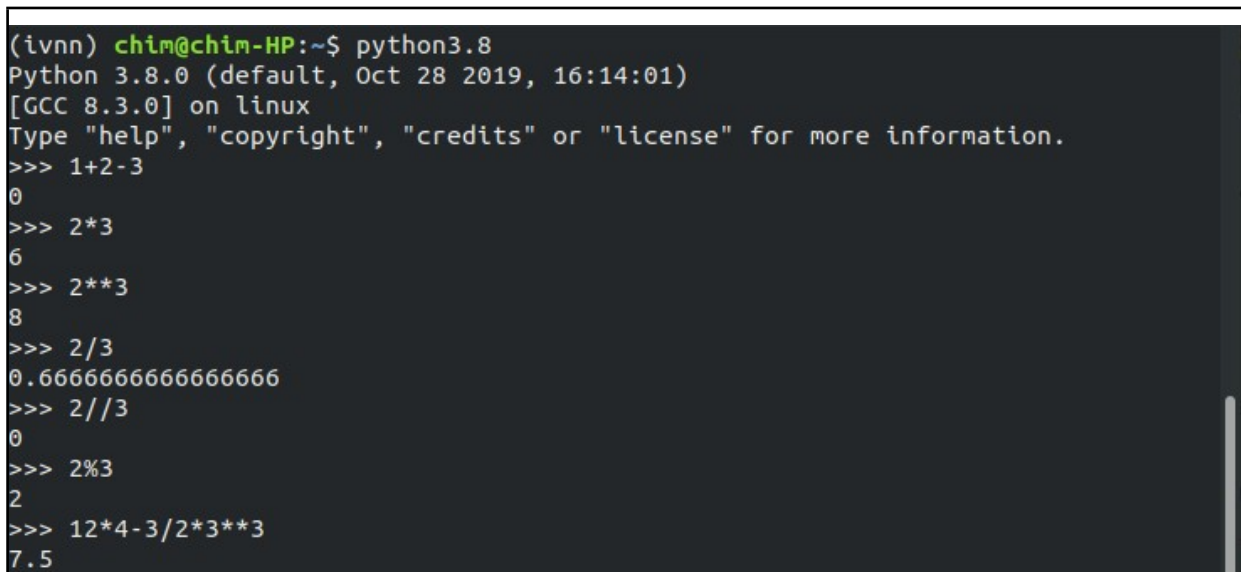
# Introduction au langage Python

## 1 Introduction

Pour lancer python, vous pouvez utiliser :

- Le shell interactif : il est généralement utilisé pour tester des fonctionnalités ou pour effectuer certains traitements instantanés.

Pour ouvrir le shell, tapez sur la ligne de commande '`python3.8`' sous Windows ou sous Linux. Le shell est aussi appelé REPL (Read, Evaluate, Print and Loop) qui consiste à lire l'instruction saisie, l'évaluer, afficher le résultat et attendre la saisie de l'instruction suivante)



```
(ivnn) chim@chim-HP:~$ python3.8
Python 3.8.0 (default, Oct 28 2019, 16:14:01)
[GCC 8.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 1+2-3
0
>>> 2*3
6
>>> 2**3
8
>>> 2/3
0.6666666666666666
>>> 2//3
0
>>> 2%3
2
>>> 12*4-3/2*3**3
7.5
```

*figure 1: REPL. Les opérateurs.*

- Création d'un fichier ayant '.py' comme extension appelé script. Il est généralement utilisé pour créer des programmes et applications. Dans ce cas, un éditeur de texte ou un IDE est utilisé pour écrire le code.
- Utiliser Jupyter qui est souvent utilisé dans les démonstrations, l'enseignement et dans la science des données (data science).

*figure 2: Interface de Jupyter*

## 2 Variables

Les variables sont utilisées pour stocker des valeurs. Dans la plupart des langages de programmation, la déclaration des variables et de leur type est obligatoire avant de les manipuler. Sous python, le type des variables est dynamique : la déclaration des variables n'est pas obligatoire, elles doivent uniquement être initialisées.

```
>>> var_1=2
>>> var_1
2
>>> 3+var_1
5
>>> var_1+=3
>>> var_1
5
>>> var_1='a'
>>> var_1
'a'
>>> var_1+var_2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'var_2' is not defined
>>> var_2=3
>>> var_1+var_2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can only concatenate str (not "int") to str
>>> var_1*var_2
'aaa'
>>> type(var_1)
<class 'str'>
>>> type(var_2)
<class 'int'>
>>> █
```

## 3 Types de données

### 3.1 Types de base :

Les valeurs des variables peuvent être de type : entier (int), réel (float) et booléen (bool). Le type booléen peut avoir une des deux valeurs possibles : True (vrai) ou False (faux).

```
>>> entier=3
>>> reel=9.5
>>> booleen=True
>>> type(entier)
<class 'int'>
>>> type(reel)
<class 'float'>
>>> type(booleen)
<class 'bool'>
>>> True and False
False
>>> True or False
True
>>> 2<3
True
>>> 2 != 5
True
>>> 5 == 5
True
>>> 5 >= 5
True
>>> █
```

Le type d'une valeur nulle (sous python : None) est :

```
>>> type(None)
<class 'NoneType'>
>>> a=None
>>> a is None
True
>>> a is not None
False
>>> █
```

### 3.2 Chaînes de caractères

Sous python, une chaîne de caractères est une suite de caractères entre ' ' ou " ". Différents traitements et opérations peuvent être effectués sur ces dernières à savoir : l'addition, la multiplication, etc :

```
>>> 'Hello world !'
'Hello world !'
>>> "Hello everyone!"
'Hello everyone!'
>>> "a"+"b"+"c"
'abc'
>>> "a"*3
'aaa'
>>> len('abcde')
5
>>> 'hello'-'o'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for -: 'str' and 'str'
>>> 'hello'*'o'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: can't multiply sequence by non-int of type 'str'
>>> 'hello'-1
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for -: 'str' and 'int'
>>> █
```

Pour plus de fonctions et méthodes, voir ce [lien](#).

### 3.3 Listes

Les listes sous python, permettent le stockage des valeurs de types différents dans une même variables :

```
>>> [1,2,3]
[1, 2, 3]
>>> ['hello', 1, None]
['hello', 1, None]
>>> type(['hello', 1, None])
<class 'list'>
>>> ['hello', 1, None]+[5, 'Hi']
['hello', 1, None, 5, 'Hi']
>>> l=[1, 2, 3, 5]
>>> l
[1, 2, 3, 5]
>>> l[0]
1
>>> len(l)
4
>>> l[-1]
5
>>> l[:2]
[1, 2]
>>> l.append(1)
>>> l
[1, 2, 3, 5, 1]
```

Pour plus de fonctions et méthodes, voir ce [lien](#).

### 3.4 Dictionnaires

Les dictionnaires consistent à attribuer pour chaque clé une valeur. Les clés appartenant au même dictionnaire doivent être uniques.

```
>>> dictionnaire= {'samedi':30, 'dimanche':33, 'lundi':29}
>>> dictionnaire
{'samedi': 30, 'dimanche': 33, 'lundi': 29}
>>> dictionnaire['mardi']=24
>>> dictionnaire
{'samedi': 30, 'dimanche': 33, 'lundi': 29, 'mardi': 24}
>>> dictionnaire.keys()
dict_keys(['samedi', 'dimanche', 'lundi', 'mardi'])
>>> dictionnaire.values()
dict_values([30, 33, 29, 24])
>>> dictionnaire['dimanche']
33
>>> █
```

Pour plus de fonctions et méthodes, voir ce .

## 4 Fonctions

Dans certains langages de programmation comme java, les accolades sont utilisées pour regrouper les instructions appartenant au même bloc : blocs des boucles, blocs de conditions et blocs de fonctions. En python, au lieu des accolades, l'indentation est utilisée.

```
>>> def puiss(x, p=2):
...     return x**p
...
>>> puiss(2)
4
>>> puiss(2,3)
8
>>> puiss(p=2,x=3)
9
>>> █
```

## 5 Conditions et boucles

Pour regrouper les instructions appartenant aux instructions conditionnelles ou répétitives l'indentation est effectuée.

## 5.1 Conditions

```
>>> l=None
>>> if l is None:
...     l=[] # init l
...
>>> l
[]
>>> x= 2.5
>>> if x<0:
...     print("x est negatif")
... elif x>0:
...     print("x est positif")
... else:
...     print("x est nul")
...
x est positif
>>> █
```

## 5.2 Boucles

```
>>> for i in range(3):
...     print(i)
...
0
1
2
>>> for i in range(3,5):
...     print(i)
...
3
4
>>> █
```

```
>>> n=20
>>> i=1
>>> s=0
>>> while i<= n:
...     s+=i
...     i+=1
...
>>> s
210
>>> i
21
>>> █
```

## 6 Bibliothèques

Python dispose un nombre important de bibliothèques (plus de 200 000 bibliothèques) parmi lesquelles nous citons :

### 6.1 Numpy

Numpy est une bibliothèque permettant la manipulation des tableaux d'une manière efficace et optimisée.

### 6.2 Scikit-learn

Scikit-learn est une bibliothèque d'apprentissage automatique. Elle est constituée d'un ensemble de modules permettant :

- le prétraitement de données,
- la sélection, l'évaluation et validation des modèles (appelés estimators),
- la création de modèles de classification, régression et clustering.

## 7 Aller plus loin

Apprendre avec un cours interactif :

- <https://www.learnpython.org/>
- <https://www.w3schools.com/python/default.asp>

Documentation python3.8 :

- <https://python.land/python-tutorial>
- <https://docs.python.org/fr/3.8/>