



# Langage et algorithmique

---

**Abdelmalek TOUMI**

[toumiab@ensta-bretagne.fr](mailto:toumiab@ensta-bretagne.fr)

8 septembre 2016

ENSTA Bretagne



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



# Déroulement

[toggle](#)[reset](#)

- Première partie : les bases de Python
- Deuxième partie : objet et suite de l'algorithmique
- Projet informatique : deux parties (sans et avec une interface graphique)



# Déroulement

[toggle](#)[reset](#)

- Première partie : les bases de Python
- Deuxième partie : objet et suite de l'algorithmique
- Projet informatique : deux parties (sans et avec une interface graphique)
  
- Programmation impérative et objet
- Notions d'architecture logicielle



# Déroulement

[toggle](#)[reset](#)

- Première partie : les bases de Python
- Deuxième partie : objet et suite de l'algorithmique
- Projet informatique : deux parties (sans et avec une interface graphique)
  
- Programmation impérative et objet
- Notions d'architecture logicielle
  
- Le cours est découpé : CM ( $\cong 30$  min) + Pause (7 min) + CM
- Utilisation des cartons (**vert**/**rouge**) pour les exercices Q/R

Plateforme : <https://moodle.ensta-bretagne.fr>



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



# Histoire des langages

[toggle](#)[reset](#)

- Début : langage machine
- Assembleur en 1950
- Fortran en 1954
- Matlab 1970
- C en 1971
- Python en 1994 (première idée en 1990)
- Java en 1995

Historique : <http://www.levenez.com>



# Histoire des langages

[toggle](#)[reset](#)

- Début : langage machine
- Assembleur en 1950
- Fortran en 1954
- Matlab 1970
- C en 1971
- Python en 1994 (première idée en 1990)
- Java en 1995

Historique : <http://www.levenez.com>

Langage choisi : Python (version 3.5)



# Langage Python

toggle

reset

- Python est interprété



# Langage Python

toggle

reset

- Python est interprété
- Python est orienté objet et permet une programmation impérative



# Langage Python

[toggle](#)[reset](#)

- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé



# Langage Python

[toggle](#)[reset](#)

- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement



# Langage Python

[toggle](#)[reset](#)

- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire



# Langage Python

[toggle](#)[reset](#)

- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche



# Langage Python

[toggle](#)[reset](#)

- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche
- Python interagit avec les programmes des autres langages



# Langage Python

[toggle](#)[reset](#)

- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche
- Python interagit avec les programmes des autres langages
- Python est indépendant de toute plate-forme



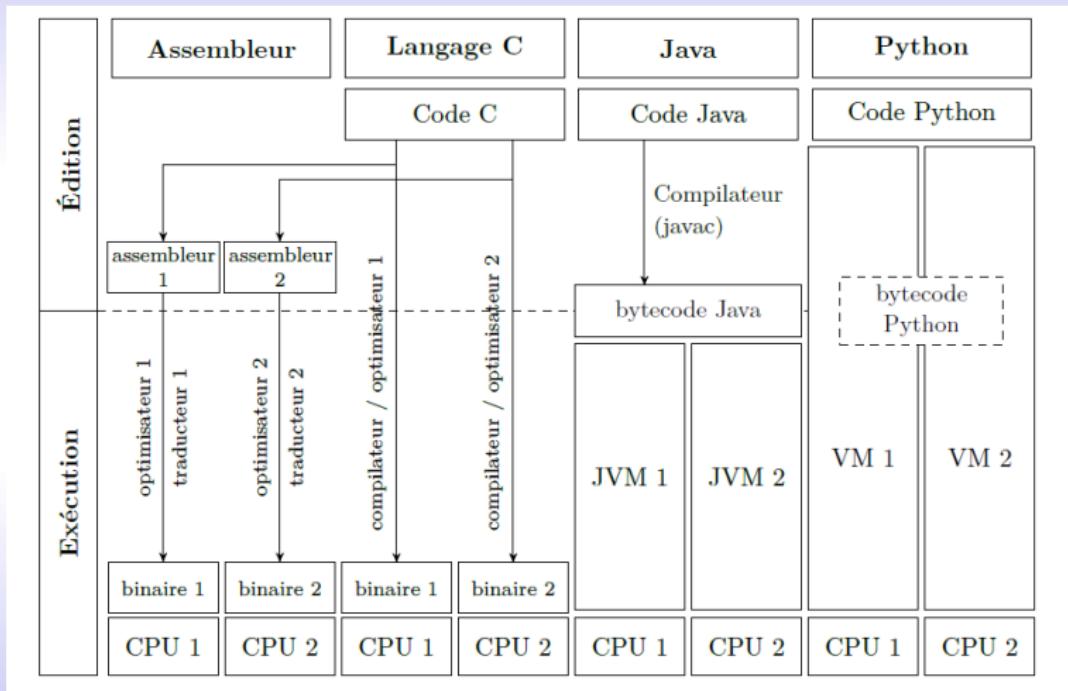
# Langage Python

[toggle](#)[reset](#)

- Python est interprété
- Python est orienté objet et permet une programmation impérative
- Python est fortement typé
- Python est typé dynamiquement
- Python assure la gestion de la mémoire
- Python est multitâche
- Python interagit avec les programmes des autres langages
- Python est indépendant de toute plate-forme
- Python est agréable et facile à lire



# Langage Python

[toggle](#)[reset](#)



# Exécution d'un programme Python

togglereset

## Remarque (Un programme Python)

Un programme est une suite d'instructions Python qui peut être exécuté de différentes manières :

- **En mode interactif**
- **En script**



# Exécution d'un programme Python

[toggle](#)[reset](#)

- En mode interactif :

- avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
  - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
  - Les instructions sont exécutées immédiatement
  - Le résultat d'une instruction est affiché sur le terminal

The screenshot shows a Windows-style application window titled "Python 3.5.1 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The main window displays the Python interpreter's prompt (>>>) followed by the command "print("bonjour")" and its output "bonjour". The status bar at the bottom indicates "GUI: OFF (TK)" and "Ln: 5 Col: 4".

```
Python 3.5.1 (v3.5.1:37a07cee5969, Dec  6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> print("bonjour")
bonjour
>>>
```



# Exécution d'un programme Python

[toggle](#)[reset](#)

- En mode interactif :

- avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
  - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
  - Les instructions sont exécutées immédiatement
  - Le résultat d'une instruction est affiché sur le terminal
- avec **ipython** : est une sourcouche de l'interpréteur Python offrant une plusieurs facilités. En version web : **IPython Notebook**

The screenshot shows a window titled "Jupyter QtConsole". The menu bar includes File, Edit, View, Kernel, Window, and Help. The main area displays an IPython session:

```
Jupyter QtConsole 4.2.1
File Edit View Kernel Window Help
Jupyter 3.5.1 (v3.5.1:37a07ceee5969, Dec 6 2015, 01:54:25) [MSC v.1900 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 4.1.2 -- An enhanced Interactive Python.
?          -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help       -> Python's own help system.
object?   -> Details about 'object', use 'object??' for extra details.

In [1]: 2+3
Out[1]: 5

In [2]: 
```



# Exécution d'un programme Python

[toggle](#)[reset](#)

- En mode interactif :

- avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
  - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
  - Les instructions sont exécutées immédiatement
  - Le résultat d'une instruction est affiché sur le terminal
- avec **ipython** : est une sourcouche de l'interpréteur Python offrant une plusieurs facilités. En version web : **IPython Notebook**

```
In [1]: a, b = 5, 3
In [2]: print(a)
5
In [3]: 
```



# Exécution d'un programme Python

togglereset

- En mode interactif :

- avec l'**interpréteur python** : via un terminal de commandes lancé avec la commande 'python3' ou via le **shell python**
  - Les instructions python à insérer après l'invite de commande (>>>), ou à l'invite de poursuit (...)
  - Les instructions sont exécutées immédiatement
  - Le résultat d'une instruction est affiché sur le terminal
- avec **ipython** : est une sourcouche de l'interpréteur Python offrant une plusieurs facilités. En version web : **IPython Notebook**
- avec **bpython** : analogue à ipython en plus léger et plus commode, il propose une coloration syntaxique sur la ligne de commande

```
>>> from decimal import *
>>> for x in xrange(0
xrange: ([start, ] stop[, step])
xrange([start,] stop[, step]) -> xrange object
Like range(), but instead of returning a list, returns an object that
generates the numbers in the range on demand. For looping, this is
slightly faster than range() and more memory efficient.
```



# Exécution d'un programme Python

toggle

reset

- **En mode script (Programming mode) :**
  - Écriture du programme Python (éditeur de texte) dans un fichier portant l'extension `.py`



# Exécution d'un programme Python

[toggle](#)[reset](#)

- **En mode script (Programming mode) :**

- Écriture du programme Python (éditeur de texte) dans un fichier portant l'extension `.py`
- Exécution
- `$ python monprogramme.py # pour windows`
- `$ python3 monprogramme.py # pour linux`



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



## Installation d'un environnement Python

toggle

reset

- Sous Windows, Python n'est pas installé par défaut.



## Installation d'un environnement Python

toggle

reset

- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés



# Installation d'un environnement Python

[toggle](#)[reset](#)

- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés
  - depuis Ubuntu 16.04 on dispose en parallèle de Python v2 et v3.5
  - Mac OS X 10.8, 10.9 et 10.10, Apple propose que Python 2.7



# Installation d'un environnement Python

togglereset

- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés
  - depuis Ubuntu 16.04 on dispose en parallèle de Python v2 et v3.5
  - Mac OS X 10.8, 10.9 et 10.10, Apple propose que Python 2.7
- Installation :
  - Windows : **WinPython 3.5.1.3** distribution spécifique à Windows et très facile à mettre en œuvre. Il intègre notamment : IPython, Spyder, NumPy, SciPy, Matplotlib, Pandas, SymPy, PIP...
  - + → <http://www.lfd.uci.edu/~gohlke/pythonlibs/>



# Installation d'un environnement Python

[toggle](#)[reset](#)

- Sous Windows, Python n'est pas installé par défaut.
- Sous GNU/Linux et Mac OS X, Python et la librairie standard sont intégrés
  - depuis Ubuntu 16.04 on dispose en parallèle de Python v2 et v3.5
  - Mac OS X 10.8, 10.9 et 10.10, Apple propose que Python 2.7
- Installation :
  - Windows : **WinPython 3.5.1.3** distribution spécifique à Windows et très facile à mettre en œuvre. Il intègre notamment : IPython, Spyder, NumPy, SciPy, Matplotlib, Pandas, SymPy, PIP...
  - + → <http://www.lfd.uci.edu/~gohlke/pythonlibs/>
  - GNU/Linux Ubuntu  $\geq 16.04$  : installation standard via les dépôts officiels Ubuntu de Canonical
  - Mac OS X : **Anaconda** distribution Python multiplateforme très répandue dans les milieux scientifiques



# Environnement de développement

togglereset

- bloc-note, notepad++, vim, emacs ;
- IDE (Integrated Development Environment) : <https://wiki.python.org/moin/IntegratedDevelopmentEnvironments>
  - Netbeans (Sun) avec Python/jPython
  - Eclipse (IBM) avec le plugin PyDev
  - **Spyder**
  - Komodo IDE
  - LiCipse avec PyDev
  - pyCharm
  - ....



# Python et le calcul scientifique

[toggle](#)[reset](#)

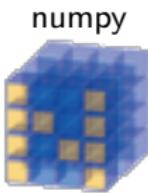
- rapide à apprendre (mais long à maîtriser) ;
- alternative (très) sérieuse à Matlab, Scilab, Octave ;
- bibliothèques de calcul et de visualisation très complète et performante ;
- parallélisation possible facilement (efficacité, etc.) ;
- communauté nombreuse et active ;
- multi-plateformes (scripts python sur Abaqus sur vos machines – Windows – et sur le cluster – Linux)



## Les principales librairies

### 1. Calcul scientifique

Pour plus d'information, consulter : <http://numpy.org/> et  
<http://scipy.org/>



- broadcasting
- multiplication de matrice
- traitement du signal
- traitement d'images
- optimisation
- interpolation
- intégration numérique
- algèbre linéaire

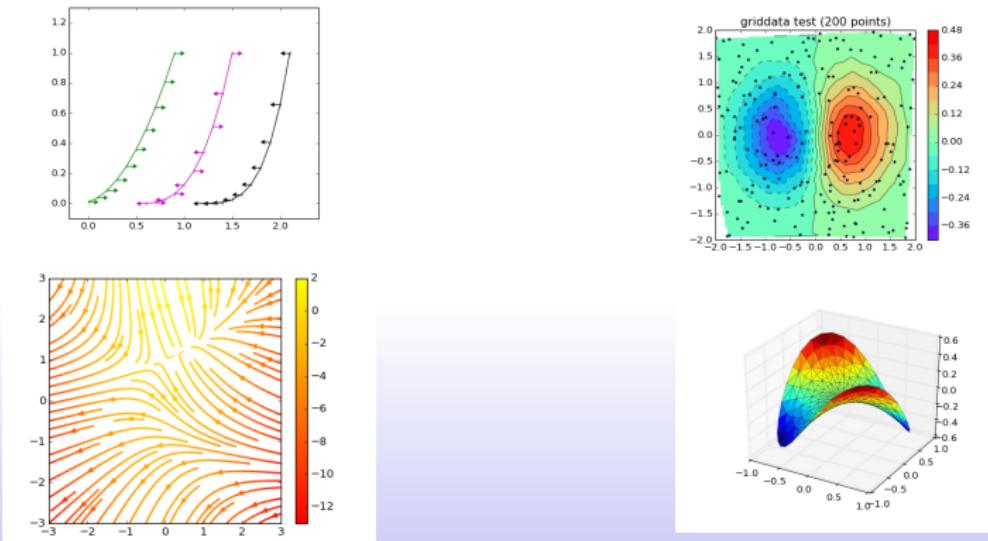


# Les principales librairies

## 2. Visualisation de données

Plus d'informations sur les sites des modules : <http://matplotlib.org/>.

matplotlib





## Quelques bibliothèques

[toggle](#)[reset](#)

- numexpr : <https://code.google.com/p/numexpr/>
- NLOpt : <http://ab-initio.mit.edu/wiki/index.php/NLOpt>
- scikits : <https://scikits.appspot.com/>
- **pyQt4** :  
<http://www.riverbankcomputing.co.uk/software/pyqt/>
- PIL : <http://www.pythonware.com/products/pil/>
- sympy : <http://sympy.org/fr/index.html>
- guizero/guidata : <https://code.google.com/p/guizero,gidata>
- pyqtgraph : <http://www.pyqtgraph.org/>
- SfePy : <http://sfepy.org/doc-devel/index.html>
- FEniCS : <http://fenicsproject.org/>



## Quelques bibliothèques

[toggle](#)[reset](#)

- numexpr : <https://code.google.com/p/numexpr/>
- NLOpt : <http://ab-initio.mit.edu/wiki/index.php/NLOpt>
- scikits : <https://scikits.appspot.com/>
- **pyQt4** :  
<http://www.riverbankcomputing.co.uk/software/pyqt/>
- PIL : <http://www.pythonware.com/products/pil/>
- sympy : <http://sympy.org/fr/index.html>
- guizero/guidata : <https://code.google.com/p/guizero,gidata>
- pyqtgraph : <http://www.pyqtgraph.org/>
- SfePy : <http://sfepy.org/doc-devel/index.html>
- FEniCS : <http://fenicsproject.org/>

WinPython : <http://www.lfd.uci.edu/~gohlke/pythonlibs/>



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



---

## Variables en Python

- Règles de nommage :
  - ne peut commencer par un chiffre ou un "\_" ("\_" est à réserver pour certaines situations)
  - peut contenir des chiffres, et "\_"
  - ne peut pas contenir de séparateur
  - ne doit pas être un mot clé (33 mots clé en Python 3)
  - peut contenir des accents, mais déconseillé
- Choisir des noms parlants (rester raisonnable)



## Variables en Python

- Règles de nommage :
  - ne peut commencer par un chiffre ou un "\_" ("\_" est à réserver pour certaines situations)
  - peut contenir des chiffres, et "\_"
  - ne peut pas contenir de séparateur
  - ne doit pas être un mot clé (33 mots clé en Python 3)
  - peut contenir des accents, mais déconseillé
- Choisir des noms parlants (rester raisonnable)

`surelyReachableObjectsWhichHaveToBeMarkedAsSuch`



## Variables et données

- Tous les éléments manipulés dans un programme python sont des objets (données).
- Une variable n'est que la référence à l'objet.
- Toute variable référence un objet d'un type bien connu (entier, réel, chaîne de caractères, ...)
- Un emplacement mémoire (adresse binaire) contient des données (binaire)
- Données de taille variable (selon le type défini : entier, réel, ...)
- Python gère ses ressources (mémoire, descripteurs de fichiers...) sans intervention du programmeur, par un mécanisme de comptage de références



## Variables et données

- Exemple :

Espace des noms (variables)



Mémoire

```
>>> a = 2
```



## Variables et données

- Exemple :

Espace des noms (variables)



Mémoire

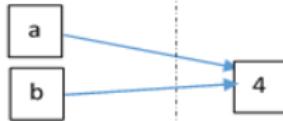
```
>>> a = 2  
>>> b = 4
```



## Variables et données

- Exemple :

Espace des noms (variables)



Mémoire

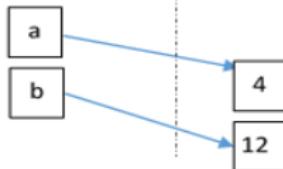
```
>>> a = 2  
>>> b = 4  
>>> a = b
```



## Variables et données

- Exemple :

Espace des noms (variables)



```
>>> a = 2  
>>> b = 4  
>>> a = b  
>>> b = 12
```



## Variables en Python

- Il n'est pas nécessaire de déclarer une variable avant de l'utiliser.
- La déclaration d'une variable et son initialisation se fait en même temps.
- Une variable peut changer de type au cours de l'exécution d'un programme.
- Noms de variables sensibles à la casse

### Exemple

```
>>> a = 42
>>> b = 33.5
>>> B = "une chaîne de caractères"
>>> a= a + 1.5
>>> a
43.5
```



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



## Fonctions utiles

- Quelques fonctions utiles intégrées à Python :

- la fonction **help()** donne l'accès à l'aide en ligne. **help(obj)** permet de consulter l'aide sur l'objet obj.
- la fonction **dir()** la liste des noms accessibles dans la portée actuelle. **dir(obj)** retourne la liste de tous les attributs valides de l'objet obj.
- la fonction **type(objet)** retourne le type de l'objet obj.
- la fonction **print(objet)** permet d'afficher le contenu de l'objet obj.

### Remarque

On invoque une fonction en utilisant l'opérateur **()** sur l'objet fonction et en plaçant les paramètres entre les parenthèses.



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



---

## Typage des variables

- Python : langage typé dynamiquement



## Typage des variables

- Python : langage typé dynamiquement
  - ⇒ typage lors de l'affectation
- Une variable peut changer de type au cours de l'exécution

### Remarque

- Pour connaître le type d'une variable ou expression, on peut utiliser la fonction **type()**
- La fonction **print()** permet d'afficher la valeur d'une variable



---

## Typage des variables

- Les types intégrés de Python



---

# Typage des variables

## • Les types intégrés de Python

- Entiers signés (complément à deux) : le type **int**
- Réels (IEEE 754) : le type **float** en double précision (64 bits)
- Complexes : le type **complex**



---

## Typage des variables

- **Les types intégrés de Python**

- Entiers signés (complément à deux) : le type **int**
- Réels (IEEE 754) : le type **float** en double précision (64 bits)
- Complexes : le type **complex**
- Booléens : le type **bool**



---

# Typage des variables

## ● Les types intégrés de Python

- Entiers signés (complément à deux) : le type **int**
- Réels (IEEE 754) : le type **float** en double précision (64 bits)
- Complexes : le type **complex**
- Booléens : le type **bool**
- Chaînes de caractères : type **str**



---

# Typage des variables

## • Les types intégrés de Python

- Entiers signés (complément à deux) : le type **int**
- Réels (IEEE 754) : le type **float** en double précision (64 bits)
- Complexes : le type **complex**
- Booléens : le type **bool**
- Chaînes de caractères : type **str**
- Listes : type **list**



---

# Typage des variables

## • Les types intégrés de Python

- Entiers signés (complément à deux) : le type `int`
- Réels (IEEE 754) : le type `float` en double précision (64 bits)
- Complexes : le type `complex`
- Booléens : le type `bool`
- Chaînes de caractères : type `str`
- Listes : type `list`
- Dictionnaires : type `dict`



---

# Typage des variables

## ● Les types intégrés de Python

- Entiers signés (complément à deux) : le type **int**
- Réels (IEEE 754) : le type **float** en double précision (64 bits)
- Complexes : le type **complex**
- Booléens : le type **bool**
- Chaînes de caractères : type **str**
- Listes : type **list**
- Dictionnaires : type **dict**
- Tuples : type **tuple**



---

# Typage des variables

## ● Les types intégrés de Python

- Entiers signés (complément à deux) : le type **int**
- Réels (IEEE 754) : le type **float** en double précision (64 bits)
- Complexes : le type **complex**
- Booléens : le type **bool**
- Chaînes de caractères : type **str**
- Listes : type **list**
- Dictionnaires : type **dict**
- Tuples : type **tuple**
- Ensembles modifiables et immuables : le type **set** et le type **frozenset**



---

## Sommaire

### 1 Introduction

Déroulement de cours

Langage Python

Installation et environnement de développement

### 2 Principes généraux

Généralités : variables et données

Fonctions utiles

### 3 Typages des variables

Les types de base

### 4 Horloge



## Horloge

---

toggle

reset

