ENSAM
ÉCOLE NATIONALE SUPÉRIEURE D'ARTS ET MÉTIERS
UNIVERSITÉ HASSAN II DE CASABLANCA

Introduction au Langage PYTHON

Filière : API-2

Dr Guelzim ibrahim

Email: ib.guelzim@gmail.com

Webographie

- https://www.docstring.fr/
- https://python.sdv.univ-paris-diderot.fr/
- https://courspython.com/
- http://www.xavierdupre.fr/app/teachpyx/helpsphinx/index.html
- https://pythontutor.com/render.html#mode=edit

Python : Introduction Générale

- Langage :
  - Gratuit
  - Interprété ≠ Compilé (C, C++,…) (**≠ence** diminue au fil du temps)
  - Faiblement typé
  - Open : code accessible
  - Indenté :

```
print("Hello")
  print("world !")
```

Message d'Erreur : 'SyntaxError: invalid syntax'

  - Multi-plateforme : portable:
    - Windows , Linux , Mac OS , Android, …
  - Ouvert aux autres langages : code python dans un code C

Introduction Générale

- Versions : 3.x
- Utilisation via un éditeur de text
  - De mieux un éditeur qui reconnait les mot clés du langage (change de couleur)
  - Atom, …
- Utilisation :
  - Plusieurs façons
    - https://www.python.org/
      - Télécharger et Installer
      - Invite de commandes (cmd) : (python + ↵) ; … Travailler … ; quit()
    - Via la plateforme Anaconda (version actuelle : 2.6.0)
    - En ligne

Début code Python

- Extension Fichier Python : .py
- Execution (invite de commandes):
  - Python nom_fich.py ↵

Début code Python

- Extension Fichier Python : .py
- Execution (invite de commandes):
  - Commentaire :
    - Une seule ligne : # (Jupyter : Ctrl + /   :: pour une ou plusieurs lignes)
    - Plusieurs lignes :

      """

      ...

      """

variables

- Moyen pour stocker de l'information
- Nom de variables : **identificateur**
  - Contient lettres, chiffres et _ (pas d'espaces ou caractères spéciaux)
  - Commence par lettre ou _
  - De préférence significatif de sa fonction :
    - prixTTC
    - age
    - salaire
  - Sensible à la casse : salaire ≠ Salaire

variables

- Moyen pour stocker de l'information
- Nom de variables : **identificateur**
  - Bonnes Pratiques :
    - PrixVente (*CamelCase*)
    - prixVente (*camelCase*)
    - prix_vente
    - Prix_Vente
- Types: int, float, str, bool
- Affectation : donner une valeur à la variable via l'opérateur =

variables : Types

- Les trois principaux types
  - entiers (integer ou int),
  - nombres décimaux (float) : le séparateur décimal est le caractère . et pas le ,
  - chaînes de caractères (string ou str).
- Aussi :
  - **Booléens** : `x = True ; y = False`
  - **Nombres complexes** : `3 + 7j ; 3 + 7J ; complex(3,7)`
  - … etc
- La fonction **type()** :

```
x = 5
print(type(x))
```

variables

- Écriture (notation) scientifique :
  - Écrire des nombres très grands ou très petits
  - Utilsant
    - des puissances de 10
    - le symbole e
- Exemple :

```
print(1e5)
print(4.15e-3)
---
100000.0
0.00415
```

variables

- Écriture (notation) scientifique :
  - AeB : $A*10^B$
  - Python génère systématiquement un float.
  - Même si le symbole e est entouré par des entiers comme dans : 1e5
- Écriture de très grands nombres:
  - Utilisation dans le code source du caractère « souligné » (ou underscore) _
  - Pour séparer des groupes de chiffres.

  Exemple (non réel):

```
Nbr_cellules = 3_301_496 # on peut aussi = 33_01_4_96
print(Nbr_cellules)
---
3301496
```

variables

- Float : attention

```
print((3 - 2.8) == 0.2)
---
False

# Solution
delta = 0.0001
var = 3.0 - 2.8
print(0.2 - delta < var < 0.2 + delta)
print (abs(var - 0.2) < delta)
```

Affichage
True True

Opérateurs / Opérations

- Opérations :
  - Affectation simple :

```
x = 5;
s = 1.2 ; y = 4 # plusieurs en une seule ligne
print(x*s) # type reel (opérateur et types opérandes)
print(x*y)
```

Affichage

```
6.0
20
```

  - Affectations séquentielles :

```
x = y = t = 2
print(x, y, t)
```

Affichage

```
2 2 2
```

  - Affectations multiples : (Expl permutation)

```
x , y = 5 , 9
print("x :",x ," - y :", y)
x,y = y,x
print("x :",x ," - y :", y)
```

Affichage

```
x : 5  - y : 9
x : 9  - y : 5
```

Opérateurs / Opérations

- Opérations sur les types numériques:
  - Arithmétiques : +, -, *

```
x = 5; s = 1.2 ; y = 4
print(x*s) # type reel (opérateur et types opérandes)
print(x*y)
---
6.0
20
```

  - Arithmétique : oprération / :: retourne par défaut un type float même si les opérandes sont des entiers

```
a = 8; b = 2
print(a/b)
---
4.0
```

Opérateurs / Opérations

- Opérations sur les types numériques:
  - Puissance : **

```
print(2**5)
```

  - Division entière : //

```
print(7//2)
```

  - Modulo (reste de la division entière) : %

```
print(7%2)
```

  - Quotient et Modulo : divmod

```
b,c = divmod(14,4)
print(b,c)
```

  - Affectation composée : += , -= , *= , /= , //= , **= , %=

```
a = 4
a **= 3
print(a)
```

Affichage

```
32
```

Affichage

```
3
```

Affichage

```
1
```

Affichage

```
3 2
```

Affichage

```
64
```

Opérateurs / Opérations

- Opérations sur les chaînes de caractères :
  - Concaténation : +

```
A = "Assalamo"
B = "Alaykom"
print(A+B)
```

Affichage
AssalamoAlaykom

  - Multiplication (par entier) : duplication : *

```
A = "Assalamo"
print(A*2)
```

Affichage
AssalamoAssalamo

  - Comportement des opérateurs : +, *
    - Différent sur les chaines de caractères
    - Notion de 'Redéfinition des opérateurs'

Opérateurs / Opérations

- Conversion (explicite) de types: Forcer une variable à changer de type, via les fonctions int(), float() et str()
  - Exemples :
    - i = 5
      str(i)

      '5'
    - i = '312'
      int(i)

      312
    - i = 456
      float(i)

      456.0
    - i = '2.3517'
      float(i)

      2.3517

Opérateurs / Opérations

- Minimum et maximum : fonctions min() et max() qui renvoient respectivement le minimum et le maximum de 2 ou plusieurs variables numériques (entiers ,floats) ou booléennes:
  - ```
    print(min(1.5, -2, 4))
    print(min(1.5, 2, 4 , False))
    print(int(min(1.5, 2, 4 , False)))
    ---
    -2
    False
    0
    ```
  - ```
    pi = 3.14
    max(1, pi , -5)
    ---
    3.14
    ```

Opérateurs / Opérations

- Saisie: input()
  - Exemple 1:

```
nomCli = input("Saisir Nom du Client : ")
print("Bonjour", nomCli)
```

  - Exemple 2:

```
prixHT = input("Prix HT du produit : ")
print("Voici le prix HT :", prixHT)
prixTTC = prixHT * 1.2
print("Voici le prix TTC :", prixTTC) # Résultat ??
```

Opérateurs / Opérations

- Conversion :

  Exemple :

```
prixHT = input("Prix HT du produit : ")
print("Voici le prix HT :", prixHT)
prixTTC = int(prixHT) * 1.2
print("Voici le prix TTC :", prixTTC) # Résultat ??
```

- Fonctions de conversion :
  - int(), float(), str(), bool()
  - Appelées aussi fonctions de **casting**

Opérateurs / Opérations

- Conversion Implicite :
  - Un opérateur binaire opère sur 2 opérandes de **même type**

```
a = 15; b = True
c = a+b
print(c)
print(type(a))
print(type(b))
print(type(c))
print(c)
```

- Conclusion : Conversion de l'opérande faible vers le type de l'opérande fort