Aide-mémoire Python 3.x[1] (version 0.6)

Exécution via deux modes : fichiers ou interpréteur embarqué (>>>…)

Déclaration et Affectation

Source : http://www.dsimb.inserm.fr/~fuchs/python/python-node3.html

En Python, la déclaration d'une variable et son initialisation (c.-à-d. la première valeur que l'on va stocker dedans) se font en même temps.

```
n = 7                          # donner à n la valeur 7
msg = "Quoi de neuf ?"         # affecter la valeur "Quoi de neuf ?" à msg
pi = 3.14159                   # assigner une valeur à la variable pi
```

Les types

Source: https://fr.wikiversity.org/wiki/Python/Les_types_de_base

Entier **int** : Entier ex : `-23`

Nombre à virgule flottante **float** : Nombre décimal ex : `7.584`

Nombre complexe **complex** ex : `2 - 3j`

Chaîne de caractères **str** ex : `"Python"` (non modifiable)

Booleen **bool** : `True` ou `False` (renvoyés par exemple lors de tests ou d'opérations booléennes)

Pour connaître le type d'une variable a : fonction `type(a)`

Les opérateurs

Source : https://fr.wikibooks.org/wiki/Programmation_Python/Op%C3%A9rateur

Arithmétiques (`+` `-` `*` `/` `**` `//` `%`) Cas part : `*=`, `+=` …

De comparaison (`>` `<` `>=` `<=` `==` `!=`)

Logiques (`or` `and` `not`)

Rem `**` exposant et `//` division entière

Les entrées-sorties

Source : http://mathsp.tuxfamily.org/spip.php?article232

interpréteur

```
>>> mot1=input()
France
>>> print(mot1)
France
>>> mot2=input("Entrez un pays : ")
Entrez un pays : Japon
>>> print(mot2)
Japon
```

fichier

```
#Curseur à la ligne :
print("Entrez un âge : ")
#Curseur à la suite :
print("Entrez un âge : ", end="")
#mot1 sera une chaîne de caractères
mot1 = intput()
#mot1 sera un nombre décimal
mot1=float(input())
#Séparateur
print("9² = ",9**2,"!", sep="")
```

Bloc d'instructions

Ligne d'en-tête terminée par deux-points + bloc d'instructions indenté par rapport à ligne d'en-tête.
Voir exemples page suivantes (structures de contrôle)

Commentaires

Sur une ligne : `# Ceci est un commentaire`

Sur plusieurs lignes :

```
""" Ceci
est
un commentaire
"""
```

1 Tutoriels et autres sur internet = faire très attention à la version de Python (au cours: version 3.x). Pour connaître les différences entre Python 2.7.x et 3.x: *http://sebastianraschka.com/Articles/2014_python_2_3_key_diff.html*

Les structures de contrôle

Source : https://fr.wikibooks.org/wiki/Programmation_Python/Structures_de_contr%C3%B4le

Les alternatives (les choix)

Forme abrégée
SI condition VRAIE → instruction
SINON rien

```
x = 5
if x == 5:
    print ("x égal 5")
...
```

Affiche : x égal 5

Forme complète [2 issues]
SI condition → instruction1
SINON instruction2

```
x = 4
if x == 5:
    print ("x égal 5")
else:
    print ("x est différent de 5")
...
```

Affiche : x est différent de 5

Forme à trois issues
SI condition1 → instruction1
SINON SI condition2 → instruction2
SINON instruction 3)

```
a = 9
if a > 10 :
    print ("a est plus grand que dix")
elif a == 10:
    print ("a est égal à dix")
else:
    print ("a est plus petit que dix")
...
```

Affiche : a est plus petit que dix

Les répétitives

Le nombre de répétitions est <u>connu</u>

```
#Bloc : n'oubliez pas le double point
#       et l'indentation
for x in range(0, 3):
    print ("C'est la", x, "e fois")
```

Le nombre de répétitions est <u>inconnu</u> mais le traitement peut ne pas être fait
(pex ici si a = 10, pas de traitement)

```
a = 0
while (a < 7):
    a = a + 1
    print (a)
```

Le nombre de répétitions est inconnu mais le traitement se fait <u>au moins une fois</u>
(break = arrêter la boucle avant la fin)

```
print("Fin si somme >100")
somme = 0
while True:
    print("Insérez un entier ", end="")
    entier = int(input())
    somme = somme + entier
    if somme > 100:
        break
print("La somme vaut : ", somme)
```

Le nombre de répétitions est inconnu mais on veut <u>éviter une valeur</u>
(continue = passer directement à l'itération suivante)

```
a = 0
while (a < 7):
    a = a + 1
    if a == 3 :
        continue
    print (a)
```

Nombres aléatoires

https://openclassrooms.com/courses/apprenez-a-programmer-en-python/un-peu-de-mathematiques

Nombre décimal compris dans [0;1[:

```
import random
alea = random.random()
```

Nombre décimal compris dans [-5;15[:

```
import random
alea = (20 * random.random()) - 5
```

Nombre entier compris dans [-5;15[:

```
import random
alea = int(20 * random.random()) - 5
```

Module Math

https://openclassrooms.com/courses/apprenez-a-programmer-en-python/un-peu-de-mathematiques
Lorsqu'une ou plusieurs fonctions du module Math sont utilisées, s'assurer que la ligne `import math` est présente avant le premier appel de fonction.

Puissance (5 exposant 4) :	`math.pow(5, 4)`
5 exposant 4	`5 ** 4`
Racine carrée (racine de 25) :	`math.sqrt(25)`
Exponentielle (e^5)	`math.exp(5)`
Valeur absolue (|-3|	`math.fabs(-3)`
Cosinus (et sinus, tangente etc → cos(1 rad))	`math.cos(1)`
Entier par excès (2,3)	`math.ceil(2.3)`
Entier par défaut (5,8)	`math.floor(5.8)`
Partie entière d'un décimal (9,5)	`math.trunc(9.5)`
Constante e	`math.e`
Constante π	`math.pi`

Un module permet donc de faire appel aux fonctions qu'il contient.

Fonctions et procédures

Source : http://python.developpez.com/cours/apprendre-python3/?page=page_9

Syntaxe Python définition fonction

```
def nomDeLaFonction(liste paramètres):
 ...
 bloc d'instructions
 ...
```

Exemple procédure (sans retour de valeur)

```
def table(base):
    n = 1
    while n <11 :
        print(n * base, end =' ')
        n = n +1
```

Appel de cette procédure

```
table(13)
```

Affiche : 13 26 39 52 65 78 91 104 117 130

Exemple de fonction (avec retour d'une val)

```
def cube(w):
    return w*w*w
...
```

Appel de cette fonction

```
b = cube(9)
print(b)
```

Affiche : 729

Voir aussi : http://www.gladir.com/CODER/PYTHON/deffonction.htm

Fonctions et procédures prédéfinies dans Python

Exemple de fonction : `lng = len("abc")` Exemple de méthode : `"abc".upper()`

Autres ex: http://apprendre-python.com/page-builtin-built-in-fonctions-internes-python (/!\ 2.7.x)

Listes (Tableaux)

Programme

```
ma_liste = [1,2,3,4]
print("ma_liste =  ", ma_liste)

ma_liste[1] = 256
print("ma_liste =  ", ma_liste)

ma_liste.append(-25)
print("ma_liste =  ", ma_liste)

ma_liste.insert(2,7)
print("ma_liste =  ", ma_liste)

ma_liste1 = ma_liste
ta_liste = [21,65,98]
print("ma_liste =  ", ma_liste)
print("ta_liste =  ", ta_liste)

ma_liste1.extend(ta_liste)
print("ma_liste  = ", ma_liste)
print("ma_liste1 = ", ma_liste1)

ma_liste = [1,7,7,7]
print("ma_liste =  ", ma_liste)
print("ma_liste1 = ", ma_liste1)

ma_liste += ta_liste
print("ma_liste =  ", ma_liste)
print("ma_liste1 = ", ma_liste1)

del ma_liste[6]
print("ma_liste =  ", ma_liste)

ma_liste.remove(7)
print("ma_liste =  ", ma_liste)
ma_liste.remove(1)
print("ma_liste =  ", ma_liste)
```

```
import random
taille = 10#déclaration d'une liste de
longueur taille
ma_liste2 = [0]*taille
print(ma_liste2)
for i in range (0,taille):
    ma_liste2[i]=int(20*random.random())-10
print("ma_liste2 = ", ma_liste2)

ma_liste3 = list()
print(ma_liste3)

liste = [1, 10, 100, 250, 500, 100]
print("Longueur liste", len(liste))
print("Combien de 100:", liste.count(100))
print("Position du 1er 100:",
liste.index(100))
print("liste = [", end="")
for nbre in liste:
    print(nbre, ",", end="")
print("]")

#copie
z = liste[:]
z[2]=88
print("z     =", z)
print("liste =", liste)
#test
print("10 dans liste?", (10 in liste))

chaine = "Il était une fois"
chaine.split(" ")
print(chaine)
liste4 = ["BN","et","les","7","n"]
" ".join(liste4)
print(liste4)
```

Exécution

```
ma_liste =   [1, 2, 3, 4]
ma_liste =   [1, 256, 3, 4]
ma_liste =   [1, 256, 3, 4, -25]
ma_liste =   [1, 256, 7, 3, 4, -25]
ma_liste =   [1, 256, 7, 3, 4, -25]
ta_liste =   [21, 65, 98]
ma_liste  =  [1, 256, 7, 3, 4, -25, 21, 65, 98]
ma_liste1 =  [1, 256, 7, 3, 4, -25, 21, 65, 98]
ma_liste =   [1, 7, 7, 7]
ma_liste1 =  [1, 256, 7, 3, 4, -25, 21, 65, 98]
ma_liste =   [1, 7, 7, 7, 21, 65, 98]
ma_liste1 =  [1, 256, 7, 3, 4, -25, 21, 65, 98]
ma_liste =   [1, 7, 7, 7, 21, 65]
ma_liste =   [1, 7, 7, 21, 65]
ma_liste =   [7, 7, 21, 65]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
ma_liste2 =  [-7, 8, -4, -7, -9, 5, 7, 8, 8, -5]
[]
```

```
Longueur liste 6
Combien de 100: 2
Position du 1er 100: 2
liste = [1 ,10 ,100 ,250 ,500 ,100 ,]
z     = [1, 10, 88, 250, 500, 100]
```

```
liste = [1, 10, 100, 250, 500, 100]
10 dans liste? True
Il était une fois
['BN', 'et', 'les', '7', 'n']
```