Aide-Mémoire Python Septembre 2018

1- Les types

type	nom	exemples
booléens	bool	False, True
entiers	int	3, -7
réels	float	3.14, 7.43e-3
chaînes	str	'salut', "l'eau"
n-uplets	tuple	1,2,3
listes	list	[1,2,3]

- Pour connaitre le type de la variable var, on tape print(type(var))
- on peut modifier le contenu d'une liste
- on ne peut pas modifier le contenu d'une chaîne ou d'un tuple.

Utilisation des variables

- Affectation signe = par exemple :

$x = 4$	la variable x prend la valeur 4 et a le même type que 4 donc int
mot="bonjour"	la variable appelée mot a pour valeur "bonjour" et est de type str (chaîne)
$y = 6/3$	Ici la variable y contient la valeur 2.0 elle est de type float.

- Lire une variable input() exemple : n =input("entrez la valeur de n")

on met en italique entre les parenthèses du input ce qu'on veut voir affiché à l'écran.
Attention : Par défaut toutes les variables définies par un input sont de type str
Si on veut travailler avec un entier a on écrira a =input("$a = ?$") puis $a = int(a)$
la dernière commande change le type de a en entier (int) on peut aussi écrire directement a =int(input("$a = ?$"))

2- Opérations mathématiques

- Addition print$(3 + 7)$
- Soustraction print$(3 - 7)$
- Multiplication print$(3 * 7)$
- Division print$(3/7)$
- Exponentiation print$(a * *2)$
- Quotient et reste dans la division euclidienne print$(7//3,7\%3)$

Les Listes

La liste peut également être indexée avec des nombres négatifs selon le modèle suivant :

```
liste          : ['girafe', 'tigre', 'singe', 'souris']
indice positif :        0        1        2         3
indice négatif :       -4       -3       -2        -1
```

Par exemple si la liste ci-contre est la liste l :
$l[0]$ donne 'girafe'
$l[-2]$ donne 'singe'

On tape :	On obtient :
l=range(10)	rien
print(l)	range(10)
print(list(l))	$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$
print(l[0])	0
print(l[3])	2
print([3 : 7])	$[2, 3, 4, 5]$
print(l[5 :])	$[4, 5, 6, 7, 8, 9]$
print(l[: 4])	$[0, 1, 2, 3]$
print(l[:])	$[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]$
print(len(l)	10 (longueur de la liste)

Remarques :
- Pour changer un terme d'une liste : $l[2] = 5$
  remplace le troisième terme de la liste par 5
- Pour ajouter un terme : l.append(8) ajoute 8 à la fin de la liste
- Pour enlever un terme d'une liste : l.remove(8)
  supprime le terme égal à 8 dans la liste
- Pour enlever un terme d'une liste : l.pop() pop fait deux choses : il enlève le dernier élément de la liste et il retourne la valeur qui a été enlevée

beaucoup d'autres fonctions au sujet des listes, voir la doc officielle Python

Les Chaînes

Fonctionnent comme les listes. Mêmes opérations à ceci près qu'on ne peut modifier le contenu d'une chaîne
mot='bonjour'
mot[3] donne 'j' mais
mot[3]='b' provoque une erreur
3*mot donne 'bonjourbonjourbonjour'
mot+' à tous' donne 'bonjour à tous' (attention aux espaces)

Les boucles

Boucle while

```
i = 1
While i >= 5 :
    print(2i)
    i = i + 1

Suite du programme...
...
```

donne :

```
2
4
6
8
10
```

Boucle for

```
for i in range(5) :
    print(3i)

Suite du programme...
...
```

donne :

```
0
3
6
9
12
```

Attention :

Ne pas oublier les deux petits points après while ou après for
Ne pas oublier le décalage de la marge (4 fois espace ou touche tabulation) qui définit tout le contenu de la boucle

LesTests

```
x = 2
if x == 2 :
    print('x vaut deux')
else :
    print('x est différent de deux')

Suite du programme...
...
```

donne : `x vaut deux`

- le test "a égale b" s'écrit en python $a == b$ (pour différencier de l'affectation)
- le test "a différent de b" s'écrit en python $a != b$
- le test "a plus grand ou égal à b" s'écrit en python $a >= b$
- la condition (ici x==2) peut être un booléen (prend la valeur True ou False) ou bien un nombre. (Tout nombre différent de 0 donne True)
- l'instruction elif permet de raccourcir une alternative multiple :

```
if note < 10 : mention = "ajourné"
elif note < 12 : mention = "passable"
elif note < 14 : mention = "assez bien"
elif note < 16 : mention = "bien"
else : mention = "très bien"
```

Les fonctions

```
def f(x) :
    return x**2+3*x-2
```

```
def felicitations() :
    for i in range(3) :
        print('bravo !')
```

```
def change(l) :
    l[0]=5
```

Les modules (ou librairies)

Ce sont des fichiers qui contiennent des fonctions qui enrichissent le programme. Exemples :

- module Maths : il contient les fonctions mathématiques usuelles

```
from math import *
b=sqrt(5)
print(b**2)
```

La première ligne permet à Python de charger en mémoire les fonctions du module maths.
Si on l'omet cela provoque une erreur

- module turtle : permet de tracer des figures avec la tortue
- module fractions : permet de calculer avec les fractions et pas en flottant
- module PIL : permet de manipuler des images
- tkinter : permet de créer des interfaces graphiques

etc...