## **Mémento Python 3**

Version originale sous licence CC4 https://perso.limsi.fr/pointal/python:memento

```
entier, flottant, booléen, chaîne
                             Types de base
   int 783 0 -192
float 9.23 0.0
 bool True False
   str "Un\nDeux"
                                 d immutables
        retour à la ligne
```

```
Conversions
                                             type (expression)
int ("15") \rightarrow 15
float ("3.14") \rightarrow 3.14
int (15.56) \rightarrow 15
                                   troncature de la partie décimale
```

```
Identificateurs
pour noms de variables,
fonctions, modules, classes...
a...zA...Z_ suivi de a...zA...Z_0...9
□ accents possibles mais à éviter
□ mots clés du langage interdits
□ distinction casse min/MAJ
      © a toto x7 y_max BigOne
      ⊗ 8y and for
```

 $multiplication \Leftrightarrow x=x*2$ 

Logique booléenne

x - = 2

x\*=5

```
Variables & affectation

    affectation ⇔ association d'un nom à une valeur

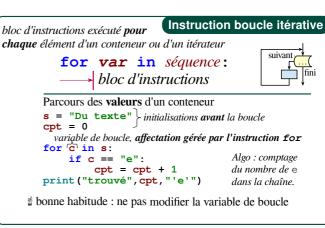
 1) évaluation de la valeur de l'expression de droite
2) affectation dans l'ordre avec les noms de gauche
x=1.2+8+sin(y)
a=b=c=0 affectation à la même valeur
y, z, r=9.2, -7.6, 0 affectations multiples
a, b=b, a échange de valeurs
                                               et
           incrémentation \Leftrightarrow x=x+3
                                               /=
                                               ,
왕=
            d\acute{e}cr\acute{e}mentation \Leftrightarrow x=x-2
```

```
Séquences d'entiers
range ([début,] fin [,pas])
           début défaut 0, fin non compris dans la séquence, pas signé et défaut 1
 range (5) \rightarrow 0 1 2 3 4
                                  range (2, 12, 3) \rightarrow 25811
 range (3,8) \rightarrow 34567
                                  range (20, 5, -5) \rightarrow 20 15 10
 d range fournit une séquence immutable d'entiers construits au besoin
```

```
Affichage
print("v=",3,"cm :",x,",
éléments à afficher : valeurs littérales, variables, expressions
Options de print:
□ sep=" "
                         séparateur d'éléments, défaut espace
□ end="\n"
                         fin d'affichage, défaut fin de ligne
                                                            Saisie
s = input("Directives:")
  input retourne toujours une chaîne, la convertir vers le type désiré
      (cf. encadré Conversions).
```

module truc⇔fichier truc.py

```
Instruction boucle conditionnelle
   bloc d'instructions exécuté
   tant que la condition est vraie
aux boucles sans
      while condition logique:
          \rightarrow bloc d'instructions
     = 0 } initialisations avant la boucle
  i = 1 condition avec au moins une valeur variable (ici i)
                                                                  Algo: i=100
  while i <= 100:
       s = s + i**2
i = i + 1
                           🖠 faire varier la variable de condition!
  print("somme:",s)
```



from monmod import nom1, nom2 as fct

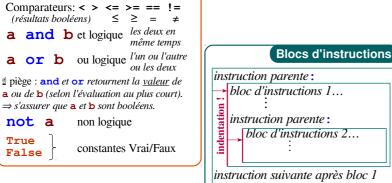
import monmod →accès via monmod.nom1 ...

if bool(x) ==False:  $\Leftrightarrow$  if not x:

Imports modules/noms

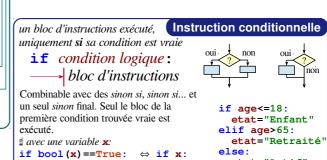
etat="Actif"

→accès direct aux noms, renommage avec as



½ régler l'éditeur pour insérer 4 espaces à

la place d'une tabulation d'indentation.



modules et packages cherchés dans le python path (cf. sys.path)

```
🕯 nombres flottants... valeurs approchées! Maths
Opérateurs : + - * / // % **
Priorités (...)
                  ÷ entière reste ÷
  d priorités usuelles
```