Mémento Bases Python 3

```
entier, flottant, booléen, chaîne, octets Types de Base
                                                     • séquences ordonnées, accès par index rapide, valeurs répétables Types Conteneurs
                                                               list [1,5,9]
                                                                                      ["x",11,8.9]
                                                                                                               ["mot"]
                                                                                                                                   [:]
   int 783 0 -192
                            0b010 0o642 0xF3
               nul
                            binaire
                                    octal
                                                            ,tuple (1,5,9)
                                                                                       11, "y", 7.4
                                                                                                               ("mot",)
                                                                                                                                   ()
float 9.23 0.0
                        -1.7e-6
                                                      bool True False
                              ×10<sup>-6</sup>
                                                             * str bytes (séquences ordonnées de caractères / d'octets)
    str "Un\nDeux"
                                                                                                                                 b""
                              Chaîne multiligne:
                                                     • conteneurs clés, sans ordre a priori, accès par clé rapide, chaque clé unique
       retour à la ligne échappé
                                """X\tY\tZ
                                1\t2\t3"""
          'L\_'âme'
                                                     dictionnaire dict {"clé":"valeur"}
                                                                                                  dict(a=3,b=4,k="v")
                                                                                                                                   {}
                                                     (couples clé/valeur) {1:"un", 3:"trois", 2:"deux", 3.14:"π"}
           ' échappé
                                tabulation échappée
bytes b"toto\xfe\775"
                                                                 set {"clé1", "clé2"}
                                                                                                   {1,9,3,0}
                                                                                                                              set()
             hexadécimal octal
                                       immutables
                                                     frozenset ensemble immutable
                                                                                                                                  vides
pour noms de variables.
                          Identificateurs
                                                                                                                        Conversions
                                                                                           type (expression)
                                             int ("15") \rightarrow 15
fonctions, modules, classes...
                                             int("3f",16) \rightarrow 63
                                                                                spécification de la base du nombre entier en 2<sup>nd</sup> paramètre
a...zA...Z_ suivi de a...zA...Z_0...9
                                             int (15.56) \rightarrow 15
                                                                                troncature de la partie décimale
□ accents possibles mais à éviter
                                             float ("-11.24e8") \rightarrow -1124000000.0
□ mots clés du langage interdits
                                             round (15.56, 1) \rightarrow 15.6
                                                                                arrondi à 1 décimale (0 décimale → nb entier)
□ distinction casse min/MAJ
                                             bool (x) False pour x nul, x conteneur vide, x None ou False ; True pour autres x
      © a toto x7 y_max BigOne
      8 8y and for
                                             str(x) → "..."
                                                                 chaîne de représentation de x pour l'affichage (cf. formatage au verso)
chr(64) \rightarrow '@'
                                                                 ord('@')→64
                                                                                             code ↔ caractère
               Affectation de Variables !
                                             repr (x) → "..." chaîne de représentation littérale de x
1) évaluation de la valeur de l'expression de droite
                                             bytes([72,9,64]) \rightarrow b'H\t@'
2) affectation dans l'ordre avec les noms de gauche
                                             list("abc") \rightarrow ['a', 'b', 'c']
 <sup>2</sup> affectation ⇔ association d'un nom à une valeur
x=1.2+8+sin(y)
                                             dict([(3, "trois"), (1, "un")]) → {1: 'un', 3: 'trois'}
a=b=c=0 affectation à la même valeur
                                             set(["un", "deux"]) → {'un', 'deux'}
y, z, r=9.2, -7.6, 0 affectations multiples
                                             str de jointure et séquence de str → str assemblée
                                                   ':'.join(['toto','12','pswd']) → 'toto:12:pswd'
a, b=b, a échange de valeurs
a, *b=seq ] dépaquetage de séquence en
                                             str découpée sur les blancs → list de str
*a, b=seq ∫ élément et liste
                                                   "des mots espacés".split() → ['des','mots','espacés']
                                             str découpée sur str séparateur → list de str
          incrémentation \Leftrightarrow x=x+3
                                       *=
                                                   "1,4,8,2".split(",") \rightarrow ['1','4','8','2']
x=2
          d\acute{e}cr\acute{e}mentation \Leftrightarrow x=x-2
                                       /=
                                             séquence d'un type → list d'un autre type (par liste en compréhension)
                                       %=
x=None valeur constante « non défini »
                                                   [int(x) for x in ('1', '29', '-3')] \rightarrow [1,29,-3]
          suppression du nom x
                                      ::::'`
                                            -----
------
                                                                                           Indexation des Conteneurs Séquences
                                      pour les listes, tuples, chaînes de caractères, bytes,...
                    -5
                                   -3
                                         -2
                                                 -1
                                                          Nombre d'éléments
                                                                                 Accès individuel aux éléments par 1st [index]
    index négatif
     index positif
                    0
                            1
                                   2
                                          3
                                                          len(lst) \rightarrow 5
                                                                                 lst[0] \rightarrow 10
                                                                                                   \Rightarrow le premier
                                                                                                                    1st[1]→20
           lst=[10,
                                  30;
                           20,
                                          40
                                                 501
                                                                                 1st [-1] → 50 \Rightarrow le dernier
                                                                                                                    1st [-2] \rightarrow 40
                                                         tranche positive
                                      3
                                             4
                                                                                 Sur les séquences modifiables (list),
                                                             (de 0 à 4 ici)
 tranche négative
                                                                                 suppression avec del lst[3] et modification
                                                                                 par affectation 1st [4] = 25
 Accès à des sous-séquences par lst [tranche début:tranche fin:pas]
                                                                                                          lst[:3] \rightarrow [10, 20, 30]
 lst[:-1] \rightarrow [10,20,30,40] lst[::-1] \rightarrow [50,40,30,20,10] lst[1:3] \rightarrow [20,30]
                                                                             lst[-3:-1] \rightarrow [30,40] lst[3:] \rightarrow [40,50]
 lst[1:-1] \rightarrow [20,30,40]
                                    lst[::-2] \rightarrow [50, 30, 10]
 lst[::2] \rightarrow [10, 30, 50]
                                    1st [:] \rightarrow [10, 20, 30, 40, 50] copie superficielle de la séquence
 Indication de tranche manquante \rightarrow à partir du début / jusqu'à la fin.
 Sur les séquences modifiables (list), suppression avec del lst[3:5] et modification par affectation lst[1:4]=[15,25]
                                       Blocs d'Instructions 'module truc⇔fichier truc.py Imports de Modules/Noms
               Logique Booléenne
Comparateurs: < >
                                                                           from monmod import nom1, nom2 as fct
                    \leq \geq
                                        instruction parente:
 (résultats booléens)
                                                                                                →accès direct aux noms, renommage avec as
                                          ▶ bloc d'instructions 1...
                                                                           import monmod →accès via monmod.nom1 ...
 a and b et logique les deux en
                        même temps
                                                                           modules et packages cherchés dans le python path (cf sys.path)
a or b ou logique l'un ou l'autre ou les deux
                                           instruction parente:
                                                                            un bloc d'instructions exécuté, Instruction Conditionnelle
                                             bloc d'instructions 2...
                                                                            uniquement si sa condition est vraie
g piège : and et or retournent la valeur de
a ou de b (selon l'évaluation au plus court).
                                                                               if condition logique:
⇒ s'assurer que a et b sont booléens.
                                                                                    bloc d'instructions
                                        instruction suivante après bloc 1
not a
              non logique
                                                                             Combinable avec des sinon si, sinon si... et
                                                                             un seul sinon final. Seul le bloc de la
                                         🖢 régler l'éditeur pour insérer 4 espaces à
                                                                                                                   if age<=18:
              constantes Vrai Faux
False
                                                                             première condition trouvée vraie est
                                         la place d'une tabulation d'indentation.
                                                                                                                     etat="Enfant"
           ------------
                                                                                                                   elif age>65:
                                                                             exécuté.
anombres flottants... valeurs approchées!
                                          angles en radians
                                                                   Maths
                                                                             ₫ avec une variable x:
                                                                                                                     etat="Retraité
                                                                             if bool(x) ==True: ⇔ if x:
                                                                                                                   else:
Opérateurs: + - * / // % **
                                        from math import sin, pi...
                                                                                                                     etat="Actif"
                                                                            if bool(x) == False: \Leftrightarrow if not x:
                                        sin(pi/4)→0.707...
Priorités (...)
                 ÷ entière reste ÷
                                        \cos(2*pi/3) \rightarrow -0.4999...
                                                                                                             Exceptions sur Erreurs
                                                                             Signalisation d'une erreur:
@ → × matricielle python3.5+numpy
                                                                               raise Exception(...)
                                        sqrt (81) →9.0
                                                                                                           Traitement des erreurs:
(1+5.3) *2→12.6
                                        log(e**2) \rightarrow 2.0
                                                                                              traitement
                                                                                                            try:
                                                                              traitement
                                                                                            erreur
traitement
                                                                                                                ▶ bloc traitement normal
abs (-3.2) \rightarrow 3.2
                                        ceil (12.5) →13
                                                                                raise
round (3.57, 1) \rightarrow 3.6
                                                                                                            except Exception as e:
                                        floor (12.5) \rightarrow 12
                                                                              normal
                                                                                           erreur
pow(4,3) \rightarrow 64.0
                                                                                                                ▶ bloc traitement erreur
                                       modules math, statistics, random,
```

decimal, fractions, numpy, etc. (cf. doc)

₫ bloc finally pour traitements finaux dans tous les cas.

d priorités usuelles

```
Instruction Boucle Conditionnelle d'bloc d'instructions exécuté pour
                                                                                                                       Instruction Boucle Itérative
    bloc d'instructions exécuté
                                                                                  chaque élément d'un conteneur ou d'un itérateur
    tant que la condition est vraie
aux boucles sans
       while condition logique:
                                                                                                 for var in séquence:
                                                                   Contrôle de Boucle
                                                          break
                                                                                                                                                     fini
                                                                         sortie immédiate
                                                                                                       bloc d'instructions
              bloc d'instructions
                                                          continue itération suivante
                                                           ¹ bloc else en sortie normale de
                                                                                            Parcours des valeurs d'un conteneur
            initialisations avant la boucle
   i = 1]
                                                           boucle.
                                                                                             s = "Du texte" | initialisations avant la boucle
            condition avec au moins une valeur variable (ici i)
                                                                                             cpt = 0
                                                                                                                                                        g
           i <= 100':
                                                                                             variable de boucle, affectation gérée par l'instruction for for c in s:
                                                                                                                                                        la variable
                                                                         \sum_{i}^{\infty} i^2
         s
              s + i**2
         i = i + 1
                                                                                                  if c == "e":
                            🖠 faire varier la variable de condition !
                                                                                                                                      Algo: comptage
   print("somme:",s)
                                                                                                        cpt = cpt + 1
                                                                                                                                      du nombre de e
    -----
                                                                                             print("trouvé", cpt, "'e'")
                                                                                                                                      dans la chaîne.
                                                                      Affichage
                                                                                   boucle sur dict/set ⇔ boucle sur séquence des clés
                                                                                                                                                        pas modifier
                             "cm :
                                                                                    utilisation des tranches pour parcourir un sous-ensemble d'une séquence
                                                                                    Parcours des index d'un conteneur séquence
 éléments à afficher : valeurs littérales, variables, expressions
                                                                                    - changement de l'élément à la position
 Options de print:
                                                                                   accès aux éléments autour de la position (avant/après)
 □ sep=" "
                                                                                                                                                        ne.
                               séparateur d'éléments, défaut espace
                                                                                   lst = [11, 18, 9, 12, 23, 4, 17]
 □ end="\n"
                               fin d'affichage, défaut fin de ligne
                                                                                    perdu = []
                                                                                                                                                        habitude
                                                                                                                                Algo: bornage des
                               print vers fichier, défaut sortie standard
 □ file=sys.stdout
                                                                                   for idx in range(len(lst)):
                                                                                                                                valeurs supérieures à 15,
                                                                                         val = lst[idx]
                                                                         Saisie
                                                                                                                                mémorisation des
  s = input("Directives:")
                                                                                          if val > 15:
                                                                                                                                valeurs perdues.
                                                                                               perdu.append(val)
     input retourne toujours une chaîne, la convertir vers le type désiré
                                                                                                                                                        ponne
                                                                                    lst[idx] = 15
print("modif:",lst,"-modif:",perdu)
         (cf encadré Conversions au recto).
                               Opérations Génériques sur Conteneurs
len (c) \rightarrow nb d'éléments
                                                                                    Parcours simultané index et valeurs de la séquence:
            max(c) sum(c)
 min(c)
                                               Note: Pour dictionnaires et ensembles,
                                                                                    for idx, val in enumerate(lst):
 sorted(c) → list copie triée
                                                ces opérations travaillent sur les clés.
 val in c → booléen, opérateur in de test de présence (not in d'absence)
                                                                                                                                Séquences d'Entiers
                                                                                      range ([début,] fin [,pas])
 enumerate (c) → itérateur sur (index, valeur)
                                                                                     début défaut 0, fin non compris dans la séquence, pas signé et défaut 1
 zip (c1, c2...) → itérateur sur tuples contenant les éléments de même index des c
                                                                                    range (5) \rightarrow 0 1 2 3 4
                                                                                                                   range (2.12.3) \rightarrow 25811
 all (c) → True si tout élément de c évalué vrai, sinon False
                                                                                    range (3, 8) \rightarrow 34567
                                                                                                                   range (20, 5, -5) \rightarrow 20 15 10
 any (c) → True si au moins un élément de c évalué vrai, sinon False
                                                                                    range (len (séq)) \rightarrow séquence des index des valeurs dans séq
 Spécifique aux conteneurs de séquences ordonnées (listes, tuples, chaînes, bytes...)
                                                                                    🛮 range fournit un séquence immutable d'entiers construits au besoin
                                     c*5→ duplication c+c2→ concaténation
 reversed (c) → itérateur inversé
                                                                                     nom de la fonction (identificateur)
                                                                                                                              Définition de Fonction
 c.index (val) \rightarrow position
                                       c. count (val) \rightarrow nb d'occurences
 import copy
                                                                                                  paramètres nommés
 copy . copy (c) → copie superficielle du conteneur
                                                                                      def fct(x,y,z):
                                                                                                                                                fct
 copy.deepcopy(c) → copie en profondeur du conteneur
                                                                                             """documentation"""
                                                      Opérations sur Listes
                                                                                            # bloc instructions, calcul de res, etc.
 nodification de la liste originale
                                                                                            return res←
                                                                                                                 - valeur résultat de l'appel, si pas de résultat
 lst.append(val)
                                 ajout d'un élément à la fin
                                                                                                                  calculé à retourner : return None
 lst.extend(seq)
                                 ajout d'une séquence d'éléments à la fin
                                                                                     lst.insert(idx, val)
                                 insertion d'un élément à une position
                                                                                      variables de ce bloc n'existent que dans le bloc et pendant l'appel à la
 lst.remove(val)
                                 suppression du premier élément de valeur val
                                                                                     fonction (penser "boite noire")
                                 supp. & retourne l'item d'index idx (défaut le dernier)
                                                                                     Avancé: def fct(x,y,z,*args,a=3,b=5,**kwargs):
 1st.pop([idx]) \rightarrow valeur
 lst.sort() lst.reverse() tri/inversion de la liste sur place
                                                                                        *args nb variables d'arguments positionnels (→tuple), valeurs par
                                                                                       \textit{d\'efaut, **kwargs nb variable d'arguments nomm\'es } (\rightarrow \texttt{dict})
    Opérations sur Dictionnaires
                                                Opérations sur Ensembles
                                           Opérateurs:
                                                                                      \mathbf{r} = \mathbf{fct}(3, \mathbf{i} + 2, 2 * \mathbf{i})
                                                                                                                                    Appel de fonction
                        d.clear()
 d[clé] =valeur
                                                                                      stockage/utilisation
                                                                                                           une valeur d'argument
                                             | → union (caractère barre verticale)
\mathbf{d}[cl\acute{e}] \rightarrow valeur
                        del d[clé]
                                                                                      de la valeur de retour par paramètre
d. update (d2) { mise à jour/ajout des couples
                                             & → intersection
                                             - ^{\wedge} \rightarrow différence/diff. symétrique
                                                                                                                                                   fct
                                                                                                                                  fct()
                                                                                    d c'est l'utilisation du nom
 d.keys()
                                                                                                                    Avancé:
d.values() \rightarrow vues itérables sur les d.items() \cdot clés / valeurs / couples
                                             < >= \rightarrow relations d'inclusion
                                                                                    de la fonction avec les
                                                                                                                    *séquence
                                           Les opérateurs existent aussi sous forme
                                                                                    parenthèses qui fait l'appel
                                                                                                                    **dict
                                           de méthodes.
d.pop (cl\acute{e}[,d\acute{e}faut]) \rightarrow valeur
                                           s.update(s2) s.copy()
                                                                                                                            Opérations sur Chaînes
d.popitem() \rightarrow (clé, valeur)
                                                                                    s.startswith(prefix[,début[,fin]])
                                           s.add(clé) s.remove(clé)
                                                                                    s.endswith(suffix[,début[,fin]]) s.strip([caractères])
d.get (clé[,défaut]) \rightarrow valeur
                                           s.discard(clé) s.clear()
                                        s.pop()
 d.setdefault (clé[,défaut]) →valeur
                                                                                    s.count(sub[,d\acute{e}but[,fin]]) s.partition(sep) \rightarrow (avant,sep,apr\grave{e}s)
                                                                                   s.index(sub[,début[,fin]]) s.find(sub[,début[,fin]])
                                                                       Fichiers :
                                                                                    s.is...() tests sur les catégories de caractères (ex. s.isalpha())
 stockage de données sur disque, et relecture
                                                                                    s.upper()
                                                                                                                     s.title() s.swapcase()
       f = open("fic.txt", "w", encoding="utf8")
                                                                                                    s.lower()
                                                                                    s.casefold()
                                                                                                         s.capitalize()
                                                                                                                               s.center([larg,rempl])
 variáble
                 nom du fichier
                                    mode d'ouverture
                                                              encodage des
                                                                                    s.ljust([larg,rempl]) s.rjust([larg,rempl]) s.zfill([larg])
                                    □ 'r' lecture (read)
                                                              caractères pour les
 fichier pour
                 sur le disque
                                                                                    s.encode (codage)
                                                                                                            s.split([sep])
                                                                                                                               s.join(séq)
                                    □ 'w' écriture (write)
                                                              fichiers textes:
 les opérations
                 (+chemin...)
                                    □ 'a' ajout (append)
□ ...'+' 'x' 'b' 't'
                                                                                      directives de formatage
                                                                                                                      valeurs à formater
                                                              utf8
                                                                      ascii
                                                                                                                                            Formatage
 cf modules os, os.path et pathlib
                                                             latin1
                                                                                     "modele{} {} {}".format(x,y,r)-
     en écriture
                                     lit chaîne vide si fin de fichier
                                                                   en lecture
                                                                                     "{sélection: formatage!conversion}"
                                   f.read([n])
                                                          .
→ caractères suivants
 f.write("coucou")
                                          si n non spécifié, lit jusqu'à la fin!
                                                                                     □ Sélection :
                                                                                                                  "{:+2.3f}".format(45.72793)
 f.writelines (list de lignes)
                                  f.readlines ([n]) \rightarrow list lignes suivantes
                                                                                                                  →'+45.728'
                                                                                                             Exemples
                                  f.readline()
                                                          → ligne suivante
                                                                                        nom
                                                                                                                  "{1:>10s}".format(8, "toto")
           🖢 par défaut mode texte 🕇 (lit/écrit str), mode binaire b
                                                                                        0.nom
                                                                                                                             toto'
                                                                                        4[clé]
                                                                                                                  "{x!r}".format(x="L'ame")
           possible (lit/écrit bytes). Convertir de/vers le type désiré!
                                                                                        0[2]
                                                                                                                  →'"L\'ame"'
 f.close()
                      ne pas oublier de refermer le fichier après son utilisation!
                                                                                     □ Formatage :
                                     f.truncate([taille]) retaillage
 f.flush() écriture du cache
                                                                                     <u>car-rempl.</u> <u>alignement signe larg.mini précision~larg.max type</u>
 lecture/écriture progressent séquentiellement dans le fichier, modifiable avec:
                                                                                               + - espace
                                                                                                              0 au début pour remplissage avec des 0
 f.tell() \rightarrow position
                                     f.seek (position[,origine])
                                                                                     entiers: b binaire, c caractère, d décimal (défaut), o octal, x ou X hexa...
 Très courant: ouverture en bloc gardé (fermeture
                                                  with open (...) as f:
                                                                                     flottant: e ou E exponentielle, f ou F point fixe, g ou G approprié (défaut),
 automatique) et boucle de lecture des lignes d'un
                                                     for ligne in f :
                                                                                     chaîne: s ...
 fichier texte:
                                                                                     □ Conversion : s (texte lisible) ou r (représentation littérale)
                                                        # traitement de ligne
```