Mémento Python 3 (partie 1)

```
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
                                                          tuple (1,5,9)
                                   octal
                                           hexa
                                                                                    11, "y", 7.4
                                                                                                           ("mot",)
                                                                                                                              ()
float 9.23 0.0
                       -1.7e-6
                                                   Valeurs non modifiables (immutables) 

    expression juste avec des virgules → tuple
 bool True False
                                                                                                                              nin
                                                          * str bytes (séquences ordonnées de caractères / d'octets)
                                                                                                                            b""
   str "Un\nDeux"
                            Chaîne multiligne:
                                                    • conteneurs clés, sans ordre a priori, accès par clé rapide, chaque clé unique
                               """X\tY\tZ
       retour à la ligne échappé
                                                  dictionnaire dict {"clé":"valeur"}
                               1\t2\t3"""
          'L\_<u>'</u>âme '
                                                                                               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
                                     ½ immutables
                                                  ₫ clés=valeurs hachables (types base, immutables...) frozenset ensemble immutable
                                                                                                                             vides
                        Identificateurs
pour noms de variables,
                                                                                        type (expression)
                                                                                                                    Conversions
```

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
   © a toto x7 y_max BigOne
                                               bool (x) False pour x nul, x conteneur vide, x None ou False; True pour autres x
                                               \operatorname{\mathtt{str}}(\mathbf{x}) \to "..." chaîne de représentation de \mathbf{x} pour l'affichage (cf. Formatage, en partie 2)
   ⊗ 8y and for
                                               chr (64) → '@' ord ('@') → 64 code \leftrightarrow core repr (x) → "..." chaîne de représentation littérale de x
                                                                                                 code ↔ caractère
                Affectation de variables
     association d'un nom à une valeur
                                               bytes([72,9,64]) \rightarrow b'H\t@'
 1) évaluation de la valeur de l'expression de droite
                                               list("abc") \rightarrow ['a', 'b', 'c']
 2) affectation dans l'ordre avec les noms de gauche
                                               dict([(3,"trois"),(1,"un")]) → {1:'un',3:'trois'}
x=1.2+8+sin(y)
                                               set(["un", "deux"]) → {'un', 'deux'}
a=b=c=0 affectation à la même valeur
y, z, r=9, -7.6, 0 affectations multiples
                                               str de jointure et séquence de str → str assemblée
                                                      ':'.join(['toto','12','pswd']) \rightarrow 'toto:12:pswd'
a, b=b, a échange de valeurs
                                               str découpée sur les blancs → list de str
"des mots espacés".split() → ['des', 'mots', 'espacés']
*a, b=seq ∫ élément et liste
                                               str découpée sur str séparateur → list de str
x+=3
           incrémentation \Leftrightarrow x=x+3
                                         et
*=
                                                     "1, \dot{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 \rightarrow 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]
del x
           suppression du nom x
```

```
pour les listes, tuples, chaînes de caractères, bytes...
                                                                                                Indexation des conteneurs séquences
                                           -2
                                                   -1
                                                             Nombre d'éléments
                                                                                      Accès individuel aux éléments par 1st [index]
    index négatif
                           -4
                                    -3
                    0
                                    2
     index positif
                            1
                                            3
                                                   4
                                                             len (1st) \rightarrow 5
                                                                                      lst[0]→10
                                                                                                         ⇒ le premier
                                                                                                                           lst[1] \rightarrow 20
                                   30;
                                           40;
                           20,
          lst=[10,
                                                   50]
                                                                                                                           1st[-2] \rightarrow 40
                                                                                      1st [-1] → 50 \Rightarrow le dernier
                                                             ₫ index à partir de 0
  tranche positive 0
                         1
                                2
                                        3
                                               4
                                                                                      Sur les séquences modifiables (list),
                                                                 (de 0 à 4 ici)
 tranche négative -5
                       -4
                               -3
                                       -2
                                               -1
                                                                                      suppression avec del lst[3] et modification
                                                                                      par affectation lst[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[1:-1] \rightarrow [20, 30, 40]
                                                                                  lst[-3:-1] \rightarrow [30,40] lst[3:] \rightarrow [40,50]
                                     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]
```

```
Logique booléenne
Comparateurs : < > <= >= !=
               ≤ ≥ =
```

(résultats booléens) **a and b** et logique les deux en même temps

a or **b** ou logique l'un ou l'autre ou les deux g piège: and et or retournent la valeur de a ou de b (selon l'évaluation au plus

⇒ s'assurer que **a** et **b** sont booléens.

not a non logique True constantes Vrai/Faux False

Blocs d'instructions

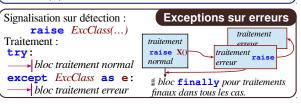
instruction parente: bloc d'instructions 1... indentation instruction parente: bloc d'instructions 2... instruction suivante après bloc 1

🖠 régler l'éditeur pour insérer 4 espaces à la place d'une tabulation d'indentation.

```
Maths
🕯 nombres flottants... valeurs approchées !
                                             angles en radians
Opérateurs : + - * / // % **
                                           from math import sin, pi...
                 ×÷
                                           \sin(pi/4) \to 0.707...
                            ↑ a<sup>b</sup>
Priorités (...)
                       Ť
                 ÷ entière reste ÷
                                           \cos(2*pi/3) \rightarrow -0.4999...
@ \rightarrow \times \text{ matricielle } (python 3.5 + numpy)
                                           sqrt (81) →9.0
                                           \log(e^{**2}) \rightarrow 2.0
(1+5.3) *2→12.6
                                           ceil(12.5)→13
abs (-3.2) →3.2
round (3.57, 1) \rightarrow 3.6
                                           floor(12.5)→12
pow(4,3) \rightarrow 64.0
                                       modules math, statistics, random
      🛮 prior<u>ités usuelles</u>
                                          decimal, fractions, numpy, etc
```

```
Modules & Imports
module truc ⇔ fichier truc.py
from monmod import nom1, nom2 as fct
   → accès direct a nom1, renommage nom2 en fct avec as
import monmod → accès via monmod.nom1 ...
modules et packages cherchés dans le python path (cf. sys.path)
```





Mémento Python 3 (partie 2)

```
Instruction boucle conditionnelle 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
                                                                                                for var in séquence:
                                               oui
      while condition logique:
                                                                   Contrôle de boucle
aux boucles
                                               non
                                                                                                                                                Г
                                                                                                       bloc d'instructions
                                                                                                                                                   fini
            \rightarrow bloc d'instructions
                                                           break
                                                                          sortie immédiate
                                                                                                                                                        boucle
                                                           continue
                                                                        itération suivante
                                                                                             Parcours des valeurs d'un conteneur
  s = 0
i = 1 initialisations avant la boucle
condition avec au moins une
                                                              ₫ bloc else en sortie
                                                                                             s = "Du texte" | initialisations avant la boucle
                                                              norma<u>le</u> de boucle.
            condition avec au moins une valeur
                                                                                             cpt = 0
                                                                                                                                                        qe
   while i <= 100:
                                variable (ici i)
                                                                                         variable de boucle, affectation gérée par l'instruction for for c in s:
                                                                Algo: i=100
                                                                                                                                                        variable
                                                                   s = \sum_{i}^{2} i^{2}
        s = s + i**2
i = i + 1
                                                                                                  if c == "e":
                                                                                                                                      Algo: comptage
                          ₫ faire varier la variable de condition!
                                                                                            cpt = cpt + 1
print("trouvé", cpt, "'e'")
   print("somme:",s)
                                                                                                                                      du nombre de e
                                                                                                                                      dans la chaîne.
                                                                                                                                                        Þ
                                                                     Affichage
                                                                                                                                                        pas modifier
                                                                                    boucle sur dict/set ⇔ boucle sur séquence des clés ; utilisation des
 print("v=", 3, "cm
                                                                                    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)
                                                                                                                                                        : ne
 □ sep=" "
                               séparateur d'éléments, défaut espace
                                                                                    lst = [11, 18, 9, 12, 23, 4, 17]
                               fin d'affichage, défaut fin de ligne
 □ end="\n"
                                                                                                                                                        habitude
                                                                                    perdu = []
                                                                                                                                Algo: bornage des
 □ file=sys.stdout
                              print vers fichier, défaut sortie standard
                                                                                    for idx in range(len(lst)):
                                                                                                                                valeurs supérieures à 15,
                                                                                         val = lst[idx]
if val > 15:
                                                                          Saisie
                                                                                                                                mémorisation des
    = input("Directives:")
                                                                                                                                valeurs perdues.
                                                                                                                                                        bonne
                                                                                              perdu.append(val)
lst[idx] = 15
    input retourne toujours une chaîne, la convertir vers le type désiré
        (cf. encadré Conversions, en partie 1).
                                                                                    print("modif:",lst,"-modif:",perdu)
                               Opérations génériques sur conteneurs
len (c) \rightarrow nb d'éléments

min (c) max (c) sum (c)

sorted (c) \rightarrow list copie triée
                                                                                    Parcours simultané index et valeurs de la séquence :
                                         Note: pour dictionnaires et ensembles,
                                                                                    for idx,val in enumerate(lst):
                                         ces opérations travaillent sur les clés.
                                                                                      range ([début, ] fin [,pas])
                                                                                                                              Séquences d'entiers
val in c → booléen, opérateur in de test de présence (not in d'absence)
                                                                                     début défaut 0, fin non compris, pas signé et défaut 1
enumerate(c) → itérateur sur (index, valeur)
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 201510
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...)
                                                                                    a range fournit une séquence immutable d'entiers construits au besoin
reversed (c) → itérateur inversé c*5→ duplication c+c2→ concaténation
                                                                                                                              Définition de fonction
                                                                                    nom de la fonction (identificateur)
c.index(val) \rightarrow position
                                    c.count (val) \rightarrow nb d'occurences
                                                                                                 paramètres nommés
import copy
copy. copy (c) \rightarrow copie superficielle (1<sup>er</sup> niveau) du conteneur
                                                                                     def fct(x,y,z):
                                                                                                                                                fct
copy. deepcopy (c) → copie en profondeur (récursive) du conteneur
                                                                                            """documentation"""
                                                                                            # bloc instructions, calcul de res, etc.

    modification de la liste originale

                                                     Opérations sur listes
                                                                                            return res ← valeur résultat de l'appel, si pas de résultat
1st.append(val)
                               ajout d'un élément à la fin
                                                                                    calculé à retourner : return None
                               ajout d'une séquence d'éléments à la fin
lst.extend(seq)
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 "boîte noire")
1st. pop ([idx]) \rightarrow valeur supp. & retourne l'item à l'index (sinon le dernier)
                                                                                     Avancé: def fct(x,y,z,*args,a=3,b=5,**kwargs):
                  lst.reverse() tri / inversion de la liste sur place
lst.sort()
                                                                                       *args \rightarrow nb variables d'arguments positionnels (tuple), a=3 \rightarrow valeurs
                                                                                      par défaut, **kwargs → nb variable d'arguments nommés (dict).
                                                Opérations sur ensembles
   Opérations sur dictionnaires
                                                                                        = fct(3, i+2, 2*i)
                                                                                                                                  Appel de fonction
                        d.clear()
d[clé] = valeur
                                          Opérateurs:
                                                                                    stockage/utilisation une valeur d'argument
                        del d[clé]
                                          | → union (caractère barre verticale)
d[cl\acute{e}] \rightarrow valeur
d. update (d2) { mise à jour/ajout des couples
                                                                                    de la valeur de retour par paramètre
                                          \mathbf{\&} \rightarrow \text{intersection}
                                                                                                                                                   fct
d c'est l'utilisation du nom de
                                                                                                                     Avancé:

    - ^ → différence/diff. symétrique

                                                                                    la fonction avec les paren-
                                          < <= > >=→ relations d'inclusion
                                                                                                                     **dict
                                                                                    thèses qui fait l'appel
                                          Existent aussi sous forme de méthodes.
d.pop(cl\acute{e}[,d\acute{e}faut]) \rightarrow valeur
                                          s.update(s2) s.copy()
                                                                                                                           Opérations sur chaînes
                                                                                    s.startswith(prefix[,début[,fin]])
d.popitem() \rightarrow (clé, valeur)
                                          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.count (sub[, d\acute{e}but[, fin]]) s.partition (sep) \rightarrow (avant, sep, apr\acute{e}s) s.index (sub[, d\acute{e}but[, fin]]) s.find (sub[, d\acute{e}but[, fin]])
d.setdefault (clé[,défaut]) →valeur
                                          s.pop()
                                                                      Fichiers
                                                                                    s.is...() tests sur les catégories de caractères (ex. s.isalpha())
s.upper() s.lower() s.title() s.swapcase()
stockage de données sur disque, et relecture
      f = open("fic.txt", "w", encoding="utf8")
                                                                                    s.casefold() s.capitalize() s.center([larg,rempl])
                nom du fichier
                                   mode d'ouverture
variable
                                                              encodage des
                                                                                    s.ljust([larg,rempl]) s.rjust([larg,rempl]) s.zfill([larg])
                                   □ 'r' lecture (read)
□ 'w' écriture (write)
                sur le disque
                                                                                                           s.split([sep]) s.join(séq)
                                                              caractères pour les
                                                                                    s.encode (codage)
fichier pour
                                                             fichiers textes :
les opérations (+ chemin...)
                                                                                    directives de formatage
cf. modules os,os.path,pathlib "...'+' 'x' 'b' 't' latin1
                                                                                                                    valeurs à formater
                                                                                     "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 :
f.writelines (list lignes)
                                                                                                                  "{:+2.3f}".format(45.72793)
                                  f.readlines ([n]) \rightarrow list lignes suiv. f.readline() \rightarrow ligne suivante
                                                                                       2
                                                                                                                 →'+45.728'
                                                                                       nom
                                                                                                                 "{1:>10s}".format(8,"toto")
                                                                                        0.nom
                                                                                                                 →' toto'
"{x!r}".format(x="L'ame")
      🖆 par défaut mode texte 🕇 (lit/écrit str), mode binaire 🕏 possible
                                                                                        4[clé]
      (lit/écrit bytes). Convertir de/vers le type désiré!
                                                                                       0[2]
                                                                                                                \rightarrow "L\ ame" '
f.close() ½ ne pas oublier de refermer le fichier après son utilisation!
                                                                                     □ Formatage :
f.flush() écriture du cache
                                    f.truncate([taille]) retaillage
                                                                                     <u>car-rempl.</u> <u>alignement signe larg.mini précision~larg.max type</u>

\stackrel{*}{\underline{}}
 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 (ferme- with open (...) as f:
                                                                                     flottant : e ou {\tt E} exponentielle, {\tt f} ou {\tt F} point fixe, {\tt g} ou {\tt G} approprié (défaut),
                                                    for ligne in f :
 ture automatique) et boucle de lecture des
                                                        # traitement de ligne
                                                                                     □ Conversion : s (texte lisible) ou r (représentation littérale)
 lignes d'un fichier texte :
```