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

```
Licence Creative Commons Paternité 4
entier, flottant, booléen, chaîne, octets Types de base
    int 783 0 -192
                             0b010 0o642 0xF3
 float 9.23 0.0
                              binaire
                                      octal
                                               hexa
                         -1.7<sub>e</sub>-6,
  bool True False
                                ×10<sup>-6</sup>
    str "Un\nDeux"
                                  Chaîne multiligne:
         retour à la ligne échappé
                                  """X\tY\tZ
                                  1\t2\t3"""
           'L<u>\</u>_âme '
             ' échappé
                                  tabulation échappée
 bytes b"toto\xfe\775"
              hexadécimal octal
                                          ½ immutables
```

Identificateurs

```
• séquences ordonnées, accès par index rapide, valeurs répétables  Types conteneurs
                          ["x",11,8.9]
         list [1,5,9]
                                                 ["mot"]
                                                                   ("mot",)
      ,tuple (1,5,9)
                             11, "y", 7.4
                                                                   ()
min.
     * str bytes (séquences ordonnées de caractères / d'octets)
                                                                  b""
• conteneurs clés, sans ordre a priori, accès par clé rapide, chaque clé unique
dictionnaire dict {"clé":"valeur"}
                                      dict(a=3,b=4,k="v")
                                                                   { }
(couples clé/valeur) {1: "un", 3: "trois", 2: "deux", 3.14: "π"}
          set {"clé1", "clé2"}
                                       {1,9,3,0}
                                                               set()
₫ clés=valeurs hachables (types base, immutables...)
                                       frozenset, ensemble immutable
                                                                  vide
```

```
© a toto x7 y_max BigOne
       8 8y and for
                  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
a, *b=seq | dépaquetage de séquence en
*a, b=seq ∫ élément et liste
x+=3
           incrémentation \Leftrightarrow x=x+3
                                             *=
x = 2
           d\acute{e}cr\acute{e}mentation \Leftrightarrow x=x-2
                                             /=
x=None valeur constante « non défini »
del x
           suppression du nom x
```

```
Conversions
int ("15") \rightarrow 15
                                          type (expression)
int("3f",16) \rightarrow 63
                                spécification de la base du nombre entier en 2<sup>nd</sup> paramètre
int (15.56) \rightarrow 15
                                troncature de la partie décimale
float ("-11.24e8") \rightarrow -1124000000.0
round (15.56, 1) \rightarrow 15.6
                                arrondi à 1 décimale (0 décimale → nb entier)
bool (x) False pour x zéro, x conteneur vide, x None ou False ; True pour autres x
str(x) \rightarrow "..." chaîne de représentation de x pour l'affichage (cf. Formatage au verso)
chr(64) \rightarrow '@' \quad ord('@') \rightarrow 64
                                            code ↔ caractère
repr (x) → "..." chaîne de représentation littérale de x
bytes([72,9,64]) \rightarrow b'H\t@'
list("abc") \rightarrow ['a', 'b', 'c']
dict([(3, "trois"), (1, "un")]) → {1: 'un', 3: 'trois'}
set(["un", "deux"]) → {'un', 'deux'}
str de jointure et séquence de str → str assemblée
     ':'.join(['toto','12','pswd']) → 'toto:12:pswd'
str découpée sur les blancs \rightarrow list de str
     "des mots espacés".split() → ['des', 'mots', 'espacés']
str découpée sur str séparateur → list de str
     "1,4,8,2".split(",") \rightarrow ['1','4','8','2']
séquence d'un type → list d'un autre type (par liste en compréhension)
     [int(x) for x in ('1', '29', '-3')] \rightarrow [1,29,-3]
```

```
Indexation conteneurs séquences
                                        pour les listes, tuples, chaînes de caractères, bytes...
                    -5
                           -4
                                   -3
                                          -2
                                                   -1
   index négatif
                                                             Nombre d'éléments
                                                                                      Accès individuel aux éléments par lst [index]
    index positif
                    0
                            1
                                    2
                                            3
                                                              len (1st) \rightarrow 5
                                                                                     lst[0]\rightarrow 10
                                                                                                         \Rightarrow le premier
                                                                                                                           1st[1]→20
          lst=[10,
                          20,
                                   30,
                                           40
                                                   501
                                                                                     1st [-1] → 50 \Rightarrow le dernier
                                                                                                                           1st[-2] \rightarrow 40
tranche positive
                        1
                                       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]

```
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
```

² piège: and et or retournent la <u>valeur</u> de a ou de b (selon l'évaluation au plus court). ⇒ s'assurer que a et b sont booléens.

non logique

```
not a
True
False
```

constantes Vrai/Faux

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

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

try:

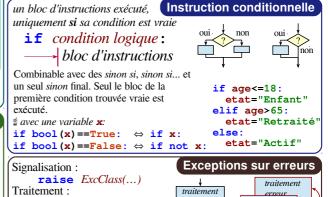
→ bloc traitement normal

→ bloc traitement erreur

except ExcClass as e:

```
angles en radians

from math import sin, pi...
sin (pi/4)→0.707...
cos (2*pi/3)→-0.4999...
sqrt (81)→9.0 
log (e**2)→2.0
ceil (12.5)→13
floor (12.5)→12
modules math, statistics, random, decimal, fractions, numpy, etc.
```



raise X()

erreur

bloc finally pour traitements

finaux dans tous les cas.

normal

```
Instruction boucle conditionnelle
                                                                                                                        Instruction boucle itérative
                                                                                    bloc d'instructions exécuté pour
   bloc d'instructions exécuté
                                                                                    chaque élément d'un conteneur ou d'un itérateur
  tant que la condition est vraie
boucles sans
                                                                                                                                                suivant
      while condition logique:
                                                                   Contrôle de boucle
                                                                                                  for var in séquence:
                                                                                                                                                  fini
             bloc d'instructions
                                                             break
                                                                           sortie immédiate
                                                                                                        bloc d'instructions
                                                             continue itération suivante
                                                                                                                                                          boucle
                                                                                              Parcours des valeurs d'un conteneur
  s = 0 initialisations avant la boucle
                                                                ₫ bloc else en sortie
anx
  i = 1 condition avec au moins une valeur variable (ici i)
                                                                normale de boucle.
                                                                                              s = "Du texte" | initialisations avant la boucle
                                                                                              cpt = 0
                                                                                                                                                          la variable de
                                                                   \overline{Algo}: i=100
  while i <= 100:
attention
                                                                                                variable de boucle, affectation gérée par l'instruction for
                                                                                              for c in s:
if c == "e":
                                                                    s = \sum_{i}^{2} i^{2}
        s = s + i**2
                            🖠 faire varier la variable de condition !
        i = i + 1
                                                                                                                                        Algo: comptage
 print("somme:",s)
                                                                                                         cpt = cpt + 1
                                                                                                                                       du nombre de \in
                                                                                              print("trouvé",cpt,"'e'")
                                                                                                                                       dans la chaîne.
                                                                      Affichage
print("v=",3,"cm :",x,",",y+4)
                                                                                     boucle sur dict/set ⇔ boucle sur séquence des clés
                                                                                                                                                          pas modifier
                                                                                     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)
                                                                                                                                                          : ne
□ sep=" "
                              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]
                                                                                                                                 mémorisation des
 s = input("Directives:")
                                                                                          if val > 15:
                                                                                                                                  valeurs perdues.
                                                                                     perdu.append(val)
lst[idx] = 15
print("modif:",lst,"-modif:",perdu)
                                                                                                                                                          ponne
    input retourne toujours une chaîne, la convertir vers le type désiré
        (cf. encadré Conversions au recto).
                              Opérations génériques sur conteneurs
                                                                                     Parcours simultané index et valeurs de la séquence :
len (c) → nb d'éléments
min(c) max(c) sum(c)
                                               Note: Pour dictionnaires et ensembles.
                                                                                     for idx, val in enumerate(lst):
sorted(c) → list copie triée
                                               ces opérations travaillent sur les clés.
                                                                                                                                Séquences d'entiers
val in c → booléen, opérateur in de test de présence (not in d'absence)
                                                                                       range ([début,] fin [,pas])
enumerate (c) \rightarrow itérateur sur (index, valeur)
zip (c1, c2...) \rightarrow itérateur sur tuples contenant les éléments de même index des c,
                                                                                     ₫ 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
all (c) → True si tout élément de c évalué vrai, sinon False
                                                                                     range (3,8) \rightarrow 3 4 5 6 7
                                                                                                                    \textbf{range (20,5,-5)} \rightarrow 20~15~10
any (c) -> True si au moins un élément de c évalué vrai, sinon False
                                                                                     range (len (s\acute{e}q)) \rightarrow s\acute{e}quence des index des valeurs dans s\acute{e}q
c.clear() supprime le contenu des dictionnaires, ensembles, listes
                                                                                     🖠 range fournit une séquence immutable d'entiers construits au besoin
Spécifique aux conteneurs de séquences ordonnées (listes, tuples, chaînes, bytes...)
reversed (c) \rightarrow itérateur inversé c*5\rightarrow duplication c+c2\rightarrow concaténation
                                                                                                                               Définition de fonction
                                                                                     nom de la fonction (identificateur)
                                     c. count (val) \rightarrow nb d'occurences
c.index (val) \rightarrow position
                                                                                                  paramètres nommés
import copy
                                                                                      def fct(x,y,z):
copy.copy (c) → copie superficielle du conteneur
                                                                                                                                                  fct
                                                                                             """documentation"""
copy.deepcopy(c) → copie en profondeur du conteneur
                                                                                             # bloc instructions, calcul de res, etc.
Opérations sur listes
                                                                                           ✓ return res 			 valeur résultat de l'appel, si pas de résultat
                                ajout d'un élément à la fin
1st.append(val)
                                                                                                                   calculé à retourner : return None
                                                                                      <sup>2</sup> les paramètres et toutes les
                                ajout d'une séquence d'éléments à la fin
lst.extend(seq)
                                                                                      variables de ce bloc n'existent que dans le bloc et pendant l'appel à la
                                insertion d'un élément à une position
lst.insert(idx, val)
                                                                                      fonction (penser "boîte noire")
lst.remove(val)
                                suppression du premier élément de valeur val
                                                                                      Avancé: def fct(x,y,z,*args,a=3,b=5,**kwargs):
1st.pop([idx]) \rightarrow valeur
                                supp. & retourne l'item d'index idx (défaut le dernier)
lst.sort() lst.reverse() tri/inversion de la liste sur place
                                                                                        *args nb variables d'arguments positionnels (→tuple), valeurs par
                                                                                        défaut, **kwargs nb variable d'arguments nommés (→dict)
  Opérations sur dictionnaires
                                                Opérations sur ensembles
                                                                                       \mathbf{r} = \mathbf{fct}(3, \mathbf{i} + 2, 2 * \mathbf{i})
                                                                                                                                     Appel de fonction
                                           Opérateurs :
                                                                                      stockage/utilisation une valeur d'argument
de la valeur de retour par paramètre
d[clé]=valeur
                        del d[clé]
                                             i → union (caractère barre verticale)
d[cl\acute{e}] \rightarrow valeur
                                               \rightarrow intersection
d. update (d2) { mise à jour/ajout des couples
                                               ^ → différence/diff. symétrique
                                                                                                                                                     fct
                                                                                                                                    fct()

    c'est l'utilisation du nom

                                                                                                                     Avancé:
d.keys()
                                             < <= > >= → relations d'inclusion
                                                                                     de la fonction avec les
                                                                                                                     *séquence
                  →vues itérables sur les
d.values () → vues itérables sur les clés / valeurs / couples d.items ()
                                           Les opérateurs existent aussi sous forme
                                                                                     parenthèses qui fait l'appel
                                                                                                                     **dict
                                           de méthodes.
d. pop (clé[,défaut]) \rightarrow valeur
                                                                                                                            Opérations sur chaînes
                                           s.update(s2) s.copy()
                                                                                     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.pop()
                                                                                     s.count(sub[,debut[,fin]]) s.partition(sep) \rightarrow (avant,sep,après)
d. setdefault (clé[,défaut]) →valeur
                                                                                     s.index(sub[,début[,fin]]) s.find(sub[,début[,fin]])
                                                                       Fichiers
stockage de données sur disque, et relecture
                                                                                     s.is...() tests sur les catégories de caractères (ex. s.isalpha())
                                                                                     s.upper() s.lower() s.title() s.swapcase()
     f = open("fic.txt", "w", encoding="utf8")
                                                                                     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])
fichier pour
                sur le disque
                                   □ 'r' lecture (read)
                                                              caractères pour les
                                                                                     s.encode (codage)
                                                                                                           s.split([sep]) s.join(séq)
                                   " w' écriture (write)
                                                              fichiers textes:
les opérations
                (+chemin...)
                                   □ 'a' ajout (append) utf8
□ ...'+' 'x' 'b' 't' latin1
                                                                     ascii
                                                                                       directives de formatage
                                                                                                                       valeurs à formater
                                                                                                                                            Formatage
cf modules os, os.path et pathlib
                                                                                      "modele{} {} {}".format(x,y,r)—
                                                                       en lecture
en écriture
                                  🖠 lit chaîne vide si fin de fichier
                                                                                      "{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 f.readline () \rightarrow ligne suivante
                                                                                        2
                                                                                                              Exemples
                                                                                                                   →'+45.728'
                                                                                                                   "{1:>10s}".format(8,"toto")

→' toto'
                                                                                        nom
                                                                                         0.nom
          🖢 par défaut mode texte 🕇 (lit/écrit str), mode binaire 🕏
                                                                                         4[clé]
                                                                                                                   "{x!r}".format(x="L'ame")
          possible (lit/écrit bytes). Convertir de/vers le type désiré!
                                                                                         0[2]
                                                                                                                  | →'"L\'ame"'
                     \sl_2 ne pas oublier de refermer le fichier après son utilisation !
f.close()
                                                                                      □ Formatage :
                                                                                      <u>car-rempl.</u> <u>alignement signe</u> <u>larg.mini .précision~larg.max</u>
f.flush() écriture du cache
                                     f.truncate ([taille]) retaillage
lecture / \'ecriture\ progressent\ s\'equentiellement\ dans\ le\ fichier,\ modifiable\ avec:
                                                                                          < > ^{\circ} = ^{\circ} + ^{\circ} 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 ...
                                                                                                                                          % pourcentage
fichier texte.
                                                        # traitement de ligne
                                                                                      □ Conversion : s (texte lisible) ou r (représentation littérale)
```