Les listes

Une liste est une structure de données qui contient une série de valeurs.
Python autorise la construction de liste contenant des valeurs de type différent (par exemple entier et chaîne de caractères), ce qui leur confère une grande flexibilité

Une liste est déclarée par une série de valeurs (*ne pas oublier les guillemets, simples ou doubles, s'il s'agit de chaînes de caractères*) séparées par des **virgules**, et le tout encadré par des **crochets**

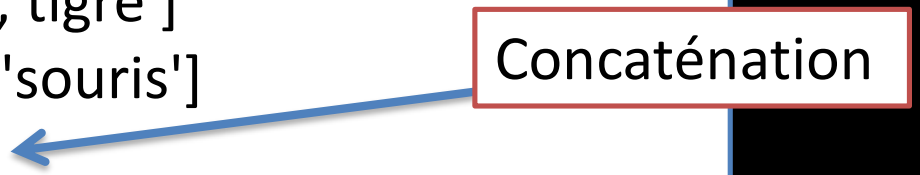

```
>>> animaux = ['girafe','tigre','singe','souris']
>>> tailles = [5, 2.5, 1.75, 0.15]
>>> mixte = ['girafe', 5, 'souris', 0.15]
>>> animaux
['girafe', 'tigre', 'singe', 'souris']
>>> tailles
[5, 2.5, 1.75, 0.15]
>>> mixte
['girafe', 5, 'souris', 0.15]
```

On appelle les éléments par leur position index ou indice ….


```
>>> animaux = ['girafe','tigre','singe','souris']
>>> animaux[0]
'girafe'
>>> animaux[1]
'tigre'
>>> animaux[3]
'souris'
```

Opérations sur les listes

```
>>> ani1 = ['girafe','tigre']
>>> ani2 = ['singe','souris']
>>> ani1 + ani2
['girafe', 'tigre', 'singe', 'souris']
>>> ani1 * 3
['girafe', 'tigre', 'girafe', 'tigre', 'girafe', 'tigre']
```



Alimentation par code

```
>>> l = [] # liste vide
>>> l
[]
>>> l = l + [15]
>>> l
[15]
>>> l = l + [-5]
>>> l
[15, -5]
```

```
>>> l = []
>>> l
[]
>>> l.append(15)
>>> l
[15]
>>> l.append(-5)
>>> l
[15, -5]
```

Indice négatif et tranches

Les indices négatifs reviennent à compter à partir de la fin. Leur principal avantage est que vous pouvez accéder au dernier élément d'une liste à l'aide de l'indice -1 sans pour autant connaître la longueur de cette liste.

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


```
>>> animaux = ['girafe','tigre','singe','souris']
>>> animaux[-1]
'souris'
>>> animaux[-2]
'singe'
>>> animaux[2]
'singe'
```

Récupérer une partie d'une liste

pour récupérer tous les éléments, du émième au énième
On utilise la notation suivante

[m:n+1]

Le n+1 nième élément n'est pas compris dans ma sélection

Exemples

Cours réalisé par le professeur Khalid 0622176552


```
>>> animaux = ['girafe', 'tigre', 'singe', 'souris']
>>> animaux[0:2]
['girafe', 'tigre']
>>> animaux[0:3]
['girafe', 'tigre', 'singe']
>>> animaux[0:]
['girafe', 'tigre', 'singe', 'souris']
>>> animaux[:]
['girafe', 'tigre', 'singe', 'souris']
>>> animaux[1:]
['tigre', 'singe', 'souris']
>>> animaux[1:-1]
['tigre', 'singe']
```

Notez que lorsqu'aucun indice n'est indiqué à gauche ou à droite du symbole :, Python prend par défaut tous les éléments depuis le début ou tous les éléments jusqu'à la fin respectivement.

On peut aussi préciser le pas en ajoutant un **:** supplémentaire et en indiquant le pas par un entier.

```
>>> animaux = ['girafe', 'tigre', 'singe', 'souris']
>>> animaux[0:3:2]
['girafe', 'singe']
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> x[::1]
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> x[::2]
[0, 2, 4, 6, 8]
>>> x[::3]
[0, 3, 6, 9]
>>> x[1:6:3]
[1, 4]
```

Finalement, on voit que l'accès au contenu d'une liste avec des crochets fonctionne sur le modèle liste **[début:fin:pas]**.

La fonction len()

L'instruction len() vous permet de connaître la longueur d'une liste, c'est-à-dire le nombre d'éléments que contient la liste.

```
>>> animaux = ['girafe', 'tigre', 'singe', 'souris']
>>> len(animaux)
4
>>> len([1, 2, 3, 4, 5, 6, 7, 8])
8
```

Générer une liste de nombres entiers

L'instruction range() va nous permettre de générer des nombres entiers compris dans un intervalle lorsqu'elle est utilisée en combinaison avec la fonction list()

```
>>> list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Lister une gamme de 10 entier

Par défaut on commence par 0

L'instruction range() fonctionne sur le modèle range **(début ,fin, pas)**. Les arguments entre crochets sont optionnels. Pour obtenir une liste, il faut l'utiliser systématiquement avec la fonction list().

```
>>> list(range(0,5))
[0, 1, 2, 3, 4]
>>> list(range(15,20))
[15, 16, 17, 18, 19]
>>> list(range(0,1000,200))
[0, 200, 400, 600, 800]
>>> list(range(2,-2,-1))
[2, 1, 0, -1]
```

```
>>> list(range(10,0,-1))
[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]
```

Listes de listes

Il est tout-à-fait possible de construire des listes de listes. Cette fonctionnalité peut être parfois très pratique
À préconiser au lieu des tableaux

```
>>> enclos1 = ['girafe', 4]
>>> enclos2 = ['tigre', 2]
>>> enclos3 = ['singe', 5]
>>> zoo = [enclos1, enclos2, enclos3]
>>> zoo
[['girafe', 4], ['tigre', 2], ['singe', 5]]
```

```
>>> zoo[1]
['tigre', 2]
```

```
>>> zoo[1][0]
'tigre'
>>> zoo[1][1]
2
```

Autres méthodes sur les listes

— append() qui permet d'ajouter un élément à la fin d'une liste existante.

```
>>> l = [1,2,3]
>>> l.append(5)
>>> l
[1, 2, 3, 5]
```

qui est équivalent à

```
>>> l = [1,2,3]
>>> l = l + [5]
>>> l
[1, 2, 3, 5]
```

— insert() pour insérer un objet dans une liste avec un indice

```
>>> l.insert(2,-15)
>>> l
[1, 2, -15, 3, 5]
```

— del pour supprimer un élément d'une liste à une indice

```
>>> del l[1]
>>> l
[1, -15, 3, 5]
```

Contrairement aux autre méthodes associées aux listes, del est une fonction générale de Python (utilisable pour d'autres objets que les listes), et celle-ci ne prend pas de parenthèse.

— remove() supprimer un élément d'une liste à partir de sa valeur.

```
>>> l.remove(5)
>>> l
[1, -15, 3]
```

— sort() pour trier une liste.

```
>>> l.sort()
>>> l
[-15, 1, 3]
```

— count() pour compter le nombre d'éléments (passé en argument) dans une liste.

```
>>> l=[1, 2, 4, 3, 1, 1]
>>> l.count(1)
3
>>> l.count(4)
1
>>> l.count(23)
0
```

attention, certaines fonctions ci-dessus décalent les indices d'une liste (par exemple insert(), del etc).

La méthode append(), que nous avons déjà vue, est particulièrement pratique car elle permet de construire une liste au fur et à mesure des itérations d'une boucle. Pour cela, nous vous rappelons qu'il est commode de définir préalablement une liste vide

```
>>> seq = 'CAAAGGTAACGC'
>>> seq_list = []
>>> seq_list
[]
>>> for base in seq:
... seq_list.append(base)
...
>>> seq_list
['C', 'A', 'A', 'A', 'G', 'G', 'T', 'A', 'A', 'C', 'G', 'C']
```

vous pouvez directement utiliser la fonction list() qui prend n'importe quel objet séquentiel (liste, chaîne de caractères, tuples .. et qui renvoie une liste (pas pratique quand on crée les élèments un à un par une boucle par exemple !!!!

```
>>> seq = 'CAAAGGTAACGC'
>>> list(seq)
['C', 'A', 'A', 'A', 'G', 'G', 'T', 'A', 'A', 'C', 'G', 'C']
```

— in permet de tester si un élément est dans la liste

```
>>> list=[1, 3, 5, 7, 9]
>>> print(3 in list)
True
```

Application

Questionnaire

1- C'est quoi une liste en Python?
2- Comment déclarer une liste?
3- Donnez la syntaxe de:
    a. Concaténation de deux listes
    b. Les 3 façons d'alimenter une liste
    c. Indiquer un élément de la liste
    d. Récupérer une partie d'une liste
    e. Retourner la longueur d'une liste
    f. Générer une liste d'entiers avec début, fin et pas
    g. Créer une liste de listes
    h. Supprimer un élément de la liste
    i. Trier une liste
    j. Compter le nombre d'occurrences d'un élément
    k. Tester si un élément est dans la liste

Application

Travaux pratiques

1- Créez une liste vide, et alimentez-la en 3 étapes on demandant à l'utilisateur son "nom" prénom" et "âge" dans cet ordre

2- Générer une liste "x" de nombres entiers entre -3 et 5
   a. Ajouter les nombres suivants à la fin de la liste -5,-8,45
   b. Ajouter les nombres suivants dans les indices correspondants 2(0), 9(5)
   c. Compter le nombre de positifs et placez-les dans une nouvelle liste listp
   d. Compter le nombre de négatifs et placez-les dans une nouvelle liste listm
   e. Tester la présence des nombres suivants
      i. 0
      ii. -4
   f. Affichez x, listep, et listm

solution page suivante

Application

```
x1=input('votre nom')

x2=input('votre prénom')

x3=input('votre age')

liste=[]                          # création de la liste obligatoire

liste.append(x1)                  # ajout de l'élément d'indice(0)

liste.append(x2)                  # ajout de l'élément d'indice(1)

liste.append(x3)                  # ajout de l'élément d'indice(2)

print (liste)
```

Application

1

```
x=list(range(-3,6))
x.append(-5)
x.append(-8)
x.append(45)
x.insert(0,88)
x.insert(5,19)
print (0 in x)
print (-4 in x)
listp=[]
listm=[]
```

2

```
for i in range (len(x)):
    if x[i]>0:
        listp.append(x[i])
    else:
        listm.append(x[i])
    print (listp)
print ("il y a " + str(len(listp))+" éléments positifs")
print(listm)
print ("il y a " + str(len(listm))+" éléments négatifs")
```