LES LISTES

Définition :

Une liste est une structure de données permettant de ranger un nombre fini d'éléments de mêmes types ou de types différents. Elle peut contenir même des listes

Création d'une liste :

Les listes en Python peuvent être créées en plaçant simplement la séquence entre crochets [ ]

```
# Liste vide
list = []
# liste avec éléments
list=['lundi','mardi']
# liste avec éléments de types différents
list=['amine','hassen', 31, 2020]
# liste entiers
list=[1, 2, 3]
# liste avec 10 éléments
list=[0] * 10
# afficher les éléments d'une liste # tous les éléments
print(list)
# premier élément
print(list[0])
# 2ème élément
print(list[1])
# dernier élément
print(list[-1])
```

Ajouter des éléments :

```
# création de la liste
list = []
# ajouter des éléments à la liste
list.append(1)
list.append(2)
list.append(4)

print("\n Liste après ajout de trois éléments ")
print(list)

# ajouter plusieurs éléments (1, 2, 3)
# avec une boucle
for i in range(1, 4):
    list.append(i)

# ajouter un élément dans une position spécifique
list.insert(0, 12)
# [12, 1, 2, 4]

# ajouter un élément dans une position spécifique
list.insert(2, 12)
# [1, 2, 12, 4]
# ajouter plusieurs éléments à la fin de la liste
List.extend([13 , 7, 9])

# [1, 2, 4, 13, 7, 9] 6 éléments dans la liste
```

Accéder aux éléments de la liste :

```
# création de la liste
List = [9, 5, 13]
# afficher les éléments d'une liste # premier élément
print(list[0])
# -> 9
# 2ème élément
print(list[1])
# -> 5
# dernier élément
print(list[-1])
# -> 13
# avant dernier élément
print(list[-2])
# -> 5
# liste imbriquée (matrice) ou tableau 2D
# 3 lignes et 4 colonnes
# 1 , 2, 5, 7
# 3, 7, 9, 8
# 10, 0, 4, 19

list=[[1, 2, 5, 7],[3, 7, 9, 8],[10, 0, 4, 19]]

# afficher l'élément de la ligne 2, 3ème colonne
print(list[1][2])
# -> 9

# afficher l'élément de la ligne 3, 2ème colonne
print(list[2][1])
# -> 0
```

Supprimer des éléments de la liste :

```
list = [5, 7, 8, 12]
# Supprimer des éléments de la liste
list.remove(5)
list.remove(8)
print("\nListe après la suppression: ")
print(list)
# -> [7, 12]

# Supprimer plusieurs éléments
for i in range(3):
    list.remove(list[i])
print("\nListe après la suppression: ")
print(list)
# -> [12]
# supprimer le dernier élément
list.pop()
print(list)
# [5, 7, 8]

# supprimer l'élément de la position 1
list.pop(1)
print(List)
# [5, 8, 12]
```

Accéder à une plage d'éléments (découpage ou slice) :

```
list = [5, 7, 8, 12, 25, 18, 30]

# afficher toutes les éléments de la liste
print(list)
# -> [5, 7, 8, 12, 25, 18, 30]

# afficher les 3 premiers éléments
print(list[:3])
# -> [5, 7, 8]

# afficher les éléments de la liste sans les 3 éléments derniers
print(list[:-3])
# -> [5, 7, 8, 12]

# afficher les éléments de la position 3 jusqu'à la fin
print(list[2:])
# -> [8, 12, 25, 18, 30]

# afficher les éléments avec un pas de 2
print(list[::2])
# -> [5, 8, 25, 30]

# afficher les éléments dans l'ordre inverse
list[::-1]
# -> [30, 18, 25, 12, 8, 7, 5]
```

Méthodes

- **sum()**
  Calcule la somme de tous les éléments de la liste.

```
1  # syntaxe : sum(List)
2  List = [1, 6, 3, 9]
3  print(sum(List))
```

- **count()**
  Calcule le nombre d'occurrence d'un élément donné de la liste.

```
1  # syntaxe : List.count(valeur)
2  List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
3  print(List.count(3))
4  # -> 2
```

- **length()**
  Calcule la taille de la liste

```
1  # syntaxe : len(liste)
2  List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
3  print(len(List))
4  # -> 9
```

- **index()**
  Retourne l'index de première occurrence. Les index de début et de fin ne sont pas des paramètres nécessaires.

```
#syntaxe : List.index(valeur[,debut[,fin]])
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
print(List.index(2))
# -> 1

# rechercher a partir de la position 3
print(List.index(2,3))
# -> 4

# rechercher dans la plage 3 - 6
print(List.index(2,3,7))
# -> 4
```

- **min()**

  Calcule le minimum de tous les éléments de la liste.

```
# syntaxe min(List)
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
print(min(List))
# -> 1
```

- **max()**

  Calcule le maximum de tous les éléments de la liste.

```
# syntaxe max(List)
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
print(max(List))
# -> 10
```

- **reverse()**

  Inverser l'ordre des éléments dans la liste

```
# syntaxe List.reverse()
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
List.reverse()
print(List)
# [2, 3, 2, 1, 2, 9, 3, 2, 10]
```

- **sort()**

  Triez (tuple et liste) par ordre croissant.

  Key et reverse_flag ne sont pas des paramètres nécessaires et reverse_flag est défini sur False, si rien n'est passé par sort().

```
# Syntaxe de sorted : sorted([list[,key[,Reverse_Flag]]])
# syntaxe de sort : list.sort([key,[Reverse_flag]])
 !:
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
List.sort(reverse=True)
# -> [10, 9, 3, 3, 2, 2, 2, 2, 1]


sorted(List)
# -> [1, 2, 2, 2, 2, 3, 3, 9, 10]
```

- **del()**

  supprimer un élément mentionné à l'aide du nom de la liste et son indice

```
# syntaxe del list[indice]
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
del List[2]
# -> List = [10, 2, 9, 2, 1, 2, 3, 2]

# supprimer une plage
del List[1:3]
# -> [10, 9, 2, 1, 2, 3, 2]
```

- **Opérateur "in"**

  Cet opérateur est utilisé pour vérifier si un élément est présent ou non dans la liste. Renvoie true si l'élément est présent dans la liste, sinon renvoie false.

```
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
if 9 in List:
    print("élément est présent dans la liste")
else:
    print("l'élément n'est pas présent dans la liste")
```

- **Opérateur "not in"**

  Cet opérateur est utilisé pour vérifier si un élément n'est pas présent dans la liste. Renvoie true si l'élément n'est pas présent dans la liste, sinon renvoie false.

```
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
if 9 not in List:
    print("l'élément n'est pas présent dans la liste")
else:
    print("l'élément est présent dans la liste")
```

- **Opérateurs "+" et "*"**

  "+" : Cet opérateur est utilisé pour concaténer deux listes en une seule liste.

  "*" : Cet opérateur est utilisé pour multiplier la liste «n» fois et renvoyer une seule liste.

```
L1 = [1, 2, 3]
L2 = [4, 5, 6]

List1=L1+L2
# -> [1,2, 3, 4, 5, 6]

List2=L1 * 3
# [1, 2, 3, 1, 2, 3, 1, 2, 3]
```

- **clear()**

  Cette fonction permet d'effacer tous les éléments de la liste. Après cette opération, la liste devient vide.

```
List = [10, 2, 3, 9, 2, 1, 2, 3, 2]
List.clear()
print(List)
# []
```