Listes

Listes

Quelques opérations élémentaires

In [1]:
```
# Concaténation
liste_a = [6156,561,56,51,651,8]
liste_b = [565,5,5,8,-58,"mpx", (1,2), bool(8),7]
liste_concat = liste_a + liste_b
print(liste_concat)
```

```
[6156, 561, 56, 51, 651, 8, 565, 5, 5, 8, -58, 'mpx', (1, 2), True, 7]
```

In [2]:
```
# Création d'une liste à partir d'un élément
liste_X = ["je veux l'X"] * 10
print(liste_X)
```

```
["je veux l'X", "je veux l'X", "je veux l'X", "je veux l'X", "je veux l'X", "je veux l'X", "je ve
ux l'X", "je veux l'X", "je veux l'X", "je veux l'X"]
```

In [3]:
```
# Longueur d'une liste
print(f'La longueur de {liste_a} est égale à', len(liste_a))
print(f'La longueur de {liste_b} est égale à', len(liste_b))
print(f'La longueur de {liste_concat} est égale à', len(liste_concat))
```

```
La longueur de [6156, 561, 56, 51, 651, 8] est égale à 6
La longueur de [565, 5, 5, 8, -58, 'mpx', (1, 2), True, 7] est égale à 9
La longueur de [6156, 561, 56, 51, 651, 8, 565, 5, 5, 8, -58, 'mpx', (1, 2), True, 7] est égale à
15
```

In [4]:
```
# Affecter à une variable la valeur d'un élément d'une liste
# La liste ne change pas !
# indice du premier élément : 0 !
prems_a = liste_a[0]
prems_b = liste_b[0]
print(f'Le premier élément de liste_a est', prems_a, liste_a)
print(f'Le premier élément de liste_b est', prems_b, liste_b)
```

```
Le premier élément de liste_a est 6156 [6156, 561, 56, 51, 651, 8]
Le premier élément de liste_b est 565 [565, 5, 5, 8, -58, 'mpx', (1, 2), True, 7]
```

In [5]:
```
#indice du dernier élément : len(liste)-1 ! ou -1 !
dern_a = liste_a[len(liste_a)-1]
dern_b = liste_b[-1]
print(f'Le dernier élément de liste_a est', dern_a, liste_a)
print(f'Le dernier élément de liste_b est', dern_b, liste_b)
```

```
Le dernier élément de liste_a est 8 [6156, 561, 56, 51, 651, 8]
Le dernier élément de liste_b est 7 [565, 5, 5, 8, -58, 'mpx', (1, 2), True, 7]
```

In [6]:
```
trois_a = liste_a[2]
trois_b = liste_b[2]
print('Le 3éme élément de liste_a est', trois_a, liste_a)
print('Le 3éme élément de liste_b est', trois_b, liste_b)
```

```
Le 3éme élément de liste_a est 56 [6156, 561, 56, 51, 651, 8]
Le 3éme élément de liste_b est 5 [565, 5, 5, 8, -58, 'mpx', (1, 2), True, 7]
```

In [ ]:
```
"""
sept_a = liste_a[6]
sept_b = liste_b[6]
print(f'Le 7éme élément de liste_a est', sept_a, liste_a)
print(f'Le 7éme élément de liste_b est', sept_b, liste_b)
"""
```

In [7]:
```
# Récupérer un "morceau" de liste : liste[m:n:pas] (n exclu)
print(liste_b, ": liste_b")

morceau_1 = liste_b[1:4]
print(morceau_1, ": morceau_1")

morceau_2 = liste_b[:len(liste_b):2] #len(liste_b) = 8
print(morceau_2, ": morceau_2")
```

```
[565, 5, 5, 8, -58, 'mpx', (1, 2), True, 7] : liste_b
[5, 5, 8] : morceau_1
[565, 5, -58, (1, 2), 7] : morceau_2
```

In [8]:
```
# Récupérer la liste en entier
liste_entier = liste_a[:]
print(liste_entier)
```

```
[6156, 561, 56, 51, 651, 8]
```

Ajouter un élément en fin de liste

In [9]:
```
# 1ére solution : on peut utiliser la commande append
liste_a_app = liste_a
print(liste_a, " : liste_a")
print(liste_a_app, " : liste_a_app")

liste_a_app.append("team mpx")
print(liste_a_app, " : liste_a_app")
# le type list est mutable donc il faut en créer une copie pour ne pas modifier l'original
print(liste_a, " : liste_a")
```

```
[6156, 561, 56, 51, 651, 8]  : liste_a
[6156, 561, 56, 51, 651, 8]  : liste_a_app
[6156, 561, 56, 51, 651, 8, 'team mpx']  : liste_a_app
[6156, 561, 56, 51, 651, 8, 'team mpx']  : liste_a
```

In [10]:
```
a = 3
b = a
b += 3
print(a)
print(b)
```

```
3
6
```

In [11]:
```
#Pour régler le problème on crée une copie de liste_a
#soit on utilise .copy()
#soit on écrit liste_a[:]
#la fonction del permet de supprimer un élément d'une liste (la taille de la liste change !)
print("longueur liste_a : ", len(liste_a))
del liste_a[-1]
print("longueur liste_a : ", len(liste_a))

liste_a_app = liste_a.copy()
print(liste_a, " : liste_a")
print(liste_a_app, " : liste_a_app")

liste_a_app.append("team mpx")
print(liste_a_app," : liste_a_app")
print(liste_a, " : liste_a")
```

```
longueur liste_a :  7
longueur liste_a :  6
[6156, 561, 56, 51, 651, 8]  : liste_a
[6156, 561, 56, 51, 651, 8]  : liste_a_app
[6156, 561, 56, 51, 651, 8, 'team mpx']  : liste_a_app
[6156, 561, 56, 51, 651, 8]  : liste_a
```

In [12]:
```
# 2éme solution : on peut utiliser la concaténation
liste_aa = liste_a
print(liste_aa, " : liste_aa")

liste_aa = liste_aa + ["team mpx"]
print(liste_aa," : liste_aa")
print(liste_a, " : liste_a")
```

```
[6156, 561, 56, 51, 651, 8]  : liste_aa
[6156, 561, 56, 51, 651, 8, 'team mpx']  : liste_aa
[6156, 561, 56, 51, 651, 8]  : liste_a
```

Remplacer un élément dans une liste

In [14]:
```
# solution : par indice
remp_a = liste_a[:]
remp_a[4] = "j'aime pythonner"
print(remp_a)
print(liste_a)
```

```
[6156, 561, 56, 51, "j'aime pythonner", 8]
[6156, 561, 56, 51, 651, 8]
```

Insérer un élément (ou plusieurs) dans une liste

In [15]:
```
# 1ére solution : on peut utiliser la fonction insert
insert_ele = liste_a[:]
print(insert_ele, ": insert_ele")
insert_ele.insert(2, "carré")
print(insert_ele, ": insert_ele")
```

```
[6156, 561, 56, 51, 651, 8] : insert_ele
[6156, 561, 'carré', 56, 51, 651, 8] : insert_ele
```

In [16]:
```
# 2éme solution : on peut utiliser la concaténation
insert_ele = liste_a[:]
print(insert_ele, ": insert_ele")
insert_ele = insert_ele[:2] + ["carré", "triangle"] + insert_ele[3:]
print(insert_ele, ": insert_ele")
```

```
[6156, 561, 56, 51, 651, 8] : insert_ele
[6156, 561, 'carré', 'triangle', 51, 651, 8] : insert_ele
```

Fonction Pop

In [17]:
```
# list.pop(indice) permet de supprimer de supprimer un élément à une position
# donnée et d'en stocker la valeur dans une variable.
# Si on ne spécifie pas la postion, le dernier élément est supprimé.

new_list = liste_a[:]
print(new_list)

new_list.pop()
print(new_list)
```

```
[6156, 561, 56, 51, 651, 8]
[6156, 561, 56, 51, 651]
```

In [18]:
```
# On peut récupérer dans une variable l'élément supprimé...
new_list = liste_a[:]
print(new_list)
removed = new_list.pop(1)
print(new_list)
print(removed)
```

```
[6156, 561, 56, 51, 651, 8]
[6156, 56, 51, 651, 8]
561
```

In [22]:
```
# ...avec la fonction del cette possibilité n'est pas possible
print(new_list)
del new_list[-1]
print(new_list)
```

```
[6156, 56, 51]
[6156, 56]
```

Manipuler une liste de listes

In [23]:
```
# On rajoute des crochets graduellement
plus_listes = [[1,2,3],[58,36,12],[256,66,58]]
douze = plus_listes[1][2]
print(douze)
```

```
12
```

In [24]:
```
# On rajoute des crochets graduellement

plus_listes = [[1,2,3],[58,36,[12,14]],[256,66,58]]
douze = plus_listes[1][2][0]

print(douze)
```

```
12
```

Trier une liste

In [25]:
```
# 1ére solution : sorted() pour un tri ascendant, sorted(reverse = False) pour un tri descendant
# VERIFIEZ BIEN QUE VOUS AVEZ LE DROIT DE L'UTILISER !
list_numbers = [561,24,3,69]
list_str = ["ronaldo","messi","fekir"]

new_list = sorted(list_numbers, reverse = True) #ascendant
new_list_str = sorted(list_str) #descendante

print(f"liste initiale : {list_numbers}, liste finale : {new_list}")
print(f"liste initiale : {list_str}, liste finale : {new_list_str}")
```

```
liste initiale : [561, 24, 3, 69], liste finale : [561, 69, 24, 3]
liste initiale : ['ronaldo', 'messi', 'fekir'], liste finale : ['fekir', 'messi', 'ronaldo']
```

In [1]:
```
# 1ére solution : list.sort() pour un tri ascendant, list.sort(reverse = True) pour un tri descenda
nt
# bien moins pratique que sorted car modifie directement la liste d'origine !
# VERIFIEZ BIEN QUE VOUS AVEZ LE DROIT DE L'UTILISER !
list_numbers = [561,24,3,69]
list_str = ["ronaldo","messi","fekir"]

print("list_str :", list_str)
print("list_numbers :", list_numbers)

new_list = list_numbers.sort()
list_str.sort(reverse = True)

print(f"list_numbers trié : {new_list}")
print(f"list_numbers trié : {list_str}")
```

```
list_str : ['ronaldo', 'messi', 'fekir']
list_numbers : [561, 24, 3, 69]
list_numbers trié : None
list_numbers trié : ['ronaldo', 'messi', 'fekir']
```