Les Listes en Python

💡 Définition et Création

Une liste est une collection de données ordonnée et modifiable, on parle de **Mutable**. Elle est reconnaissable par les crochets **[]** qui l'entourent. On peut y stocker des éléments de types différents (nombres, chaînes de caractères, booléens, etc.).

```
root@python:~$
```

```
# Création une liste vide
ma_liste = []
# Création une liste avec des éléments
fruits = ["pomme", "banane", "orange"]
# Obtenir la longueur d'une liste avec len()
longueur = len(fruits)
print(f"La liste 'fruits' contient {longueur} éléments.")
# Affiche "La liste 'fruits' contient 3 éléments."
```

⚙️ Modification de la liste

Ajouter des éléments

append(element) : ajoute un élément à la fin de la liste.
insert(index, element) : insère un élément à un index précis.
extend(iterable) : ajoute tous les éléments d'un autre objet (comme une autre liste) à la fin.

```
root@python:~$
```

```
jours = ["lundi", "mardi"]
jours.append("jeudi")
print(jours) # ['lundi', 'mardi', 'jeudi']
jours.insert(2, "mercredi")
print(jours) # ['lundi', 'mardi', 'mercredi', 'jeudi']
week_end = ["vendrei", "samedi"]
jours.extend(week_end)
print(jours)
# ['lundi', 'mardi', 'mercredi', 'jeudi', 'vendredi', 'same
```

Supprimer des éléments

pop(index) : supprime l'élément à l'index donné et le renvoie. Si l'index est omis, il supprime et renvoie le dernier élément.

```
root@python:~$
```

```
nombres = [10, 20, 30, 40]
# Supprimer le dernier élément et le stocker
element_supprime = nombres.pop()
print(f"Élément supprimé : {element_supprime}")
# Élément supprimé : 40
print(nombres)    # [10, 30]
```

🤔 Copier une liste

⚠️ Si vous attribuez une liste à une autre variable avec le signe =, vous ne faites pas une copie ! Les deux variables pointent vers le même objet en mémoire. On appelle cela une référence.

Pour faire une vraie copie, on peut utiliser la technique du slicing ou la méthode **.copy()**.

```
root@python:~$
```

```
liste_a = [1, 2, 3]
liste_b = liste_a  # C'est une référence, pas une copie !
liste_b.append(4)
print(liste_a) # Affiche [1, 2, 3, 4] !
liste_c = liste_a[:] # Vraie copie avec slicing
liste_c.append(5)
print(liste_a) # Affiche [1, 2, 3, 4]
print(liste_c) # Affiche [1, 2, 3, 4, 5]
```

🧭 Indexation et Accès aux éléments

Chaque élément d'une liste possède une position, appelée index. En Python, ⚠️ **l'indexation commence à 0**.

On peut aussi utiliser des index négatifs pour accéder aux éléments à partir de la fin de la liste. L'index -1 correspond au dernier élément, -2 à l'avant-dernier, et ainsi de suite.

```
root@python:~$
```

```
animaux = ["chien", "chat", "oiseau", "poisson"]
# Indexation positive
print(animaux[0])   # Affiche "chien"
# Indexation négative
print(animaux[-1])  # Affiche "poisson"
# Le slicing : Extraire une partie de la liste [début:fin]
# ⚠️ La fin est exclue !
print(animaux[1:3]) # Affiche ['chat', 'oiseau']
print(animaux[2:])  # Affiche ['oiseau', 'poisson']
```

📜 Parcourir une liste

La boucle for est l'outil principal pour parcourir une liste.

1. Par élément :

C'est la méthode la plus simple et la plus courante. On parcourt directement chaque valeur de la liste.

```
root@python:~$
```

```
pizzas = ["reine", "calzone"]
for pizza in pizzas:
  print(pizza)
```

```
Affiche :
reine
calzone
```

2. Par index :

Utile si vous avez besoin de la position de l'élément en plus de sa valeur. On utilise la fonction range(len(liste)).

```
root@python:~$
```

```
pizzas = ["reine", "calzone"]
for i in range(len(pizzas)):
  print(f"Pizza n°{i} : {pizzas[i]}")
```

```
Affiche :
Pizza n°0 : reine
Pizza n°1 : calzone
```

3. Parcours mixte (avec enumerate) :

La méthode la plus élégante pour avoir à la fois l'index et l'élément. Elle est à privilégier par rapport à la méthode précédente.

```
root@python:~$
```

```
pizzas = ["reine", "calzone"]
for index, pizza in enumerate(pizzas):
  print(f"Index {index} correspond à la {pizza}.")
```

```
Affiche :
Index 0 correspond à la reine.
Index 1 correspond à la calzone.
```

def decr
def decryot(key)
192.168
FOLLOW THE PYTHONIC WAY
#ACK AGAINST MACHISM