

Les listes



Définition d'une liste :

- Une **liste** est un **type de donnée** qui permet de stocker une collection d'éléments **ordonnés** et **modifiables**. Contrairement à d'autres structures de données comme les tuples ou les chaînes de caractères, les listes sont flexibles, c'est-à-dire que tu peux ajouter, modifier ou supprimer des éléments à tout moment.

Syntaxe



- Syntaxe de base pour créer une liste en Python.

```
ma_liste = [1, 2, 3, 4, 5]
```

- Une liste peut contenir différents types de données (entiers, chaînes de caractères, booléens, etc.)

```
ma_liste = [1, "ip", True, 3.14]
```



Accès aux éléments d'une liste

- Accès par l'index (commence à 0).

```
print(ma_liste[0]) # Affiche le premier élément
```

- Indexation négative pour accéder à des éléments depuis la fin de la liste.

```
print(ma_liste[-1]) # Affiche le dernier élément
```



Modifier une liste

- Modifier un élément via son index.

```
iot_devices = ["Caméra - Actif", "Thermostat - Inactif", "Capteur de mouvement - Actif"]  
iot_devices[1] = "Thermostat - Actif"  
print(iot_devices)
```



Modifier une liste

- Ajout d'éléments avec **append()** et **insert()**

```
logs_securite = [  
    "Tentative de connexion échouée",  
    "Accès non autorisé détecté",  
    "Scan de port détecté"  
]  
  
logs_securite.append("DDoS détecté")  
logs_securite.insert(1, "Malware critique détecté")
```



Supprimer un élément

- Supprimer des éléments avec **remove()** et **pop()**.

```
# Liste des adresses IP bloquées par le pare-feu
ip_bloquees = ["192.168.1.10", "10.0.0.5", "172.16.0.8", "192.168.1.15"]

# Supprimer une adresse IP spécifique si elle n'est plus considérée comme une menace
ip_bloquees.remove("10.0.0.5")

# Supprimer l'adresse IP à la position 2 (index 2)
ip_bloquees.pop(2)
```

Parcourir une liste avec la boucle For



- Utiliser une boucle **for** pour parcourir chaque élément de la liste

```
ma_liste = ["192.168.1.10", "10.0.0.5", "172.16.0.8", "192.168.1.15"]  
  
for element in ma_liste:  
    print(element)
```

Parcourir une liste avec la boucle while



- Exemple d'utilisation de la boucle **while** pour parcourir chaque élément de la liste

```
# Liste des adresses IP suspectes
ips_suspectes = ["192.168.0.100", "10.0.0.45", "172.16.0.7", "192.168.1.200"]

index = 0
while index < len(ips_suspectes):
    print(f"Analyse de l'adresse IP : {ips_suspectes[index]}")
    index += 1
```




Obtenir la taille d'une liste

- Utiliser `len()` pour obtenir la taille d'une liste

```
print(len(ma_liste))
```

Supprimer tous les éléments d'une liste



- La méthode **clear()** permet de supprimer tous les éléments d'une liste, la réinitialisant complètement sans la supprimer.

```
logs_securite = ["Tentative d'intrusion", "Accès non autorisé", "DDoS détecté"]

# Réinitialiser la liste des logs
logs_securite.clear()

# Afficher la liste vide
print(logs_securite)  # Affiche []
```

Trouver l'index d'un élément d'une liste



- La méthode **index()** renvoie la position (index) de la première occurrence d'un élément dans une liste. Si l'élément n'existe pas, une erreur est levée.

```
ips_suspectes = ["192.168.1.1", "10.0.0.2", "172.16.0.3", "192.168.1.5"]  
  
# Trouver l'index de l'IP "172.16.0.3"  
position = ips_suspectes.index("172.16.0.3")  
  
print(f"L'IP est à l'index {position}") # Affiche : L'IP est à l'index 2
```

Compter les occurrences d'un élément



- La méthode **count()** renvoie le nombre de fois qu'un élément apparaît dans une liste.

```
# Liste des tentatives de connexion avec les adresses IP
tentatives_connexion = [
    "192.168.0.10", "10.0.0.5", "192.168.0.10", "172.16.0.8",
    "192.168.0.10", "192.168.1.50", "10.0.0.5", "192.168.0.10"
]

nb_tentatives_ip = tentatives_connexion.count("192.168.0.10")

print(f"L'adresse IP 192.168.0.10 a tenté de se connecter {nb_tentatives_ip} fois.")
```