Langage Python

Les bases du langage 3 : Les listes et les tuples

Formation POEC Cybersécurité
Théo Hubert

Syntaxe

- *Listes*

Une liste est une **structure de données** qui contient une série de valeurs. Une liste est déclarée par une série de valeurs séparées par des **virgules**, et le tout encadré par des crochets [].

```
animaux = ["chiens", "chats", "brebis", "dromadaires", "souris", "serpents", "baleines"]

notes = [16, 18, 20, 20, 16, 15, 18, 19, 20]

print(type(animaux))
```

```
<class 'list'>
```

Remarque : Python autorise la construction de liste contenant des valeurs de types différents.

```
[1]  liste = ["bonjour", "aurevoir", True, False, 10, 20, 4.5, 5.5]
     print(liste)
```

```
['bonjour', 'aurevoir', True, False, 10, 20, 4.5, 5.5]
```

Syntaxe

- *Listes*

On peut facilement obtenir un élément dans une liste à l'aide de sa position (index). En programmation les listes commence par l'index 0 et termine par l'index n-1 (n étant la taille de la liste)

```
#liste  : ["girafe", "tigre", "singe", "souris"]
#indice :        0        1        2         3
```

```
animaux = ["chiens", "chats", "brebis", "baleines"]

print(animaux[0])
print(animaux[1])
print(animaux[2])
print(animaux[3])
```

```
chiens
chats
brebis
baleines
```

Syntaxe

- *Listes*

Pour ajouter un élément dans une liste, on peut utiliser la fonction append()
=> Ajoute l'élément à la fin de la liste

```
animaux = ["chiens", "chats", "brebis", "baleines"]
animaux.append("giraffe")
print(animaux)
```

```
['chiens', 'chats', 'brebis', 'baleines', 'giraffe']
```

Autrement on peut aussi utiliser la fonction insert()
=> Ajoute l'élément à la position de notre choix

```
animaux = ["chiens", "chats", "brebis", "baleines"]
animaux.insert(0, "cerf")
print(animaux)
```

```
['cerf', 'chiens', 'chats', 'brebis', 'baleines']
```

Syntaxe

- *Listes*

Pour supprimer un élément d'une liste, on peut utiliser la fonction pop()
=> Supprime un élément de la liste en fonction de sa position

```
animaux = ["chiens", "chats", "brebis", "baleines"]
animaux.pop(0)
print(animaux)
```

```
['chats', 'brebis', 'baleines']
```

Pour supprimer un élément d'une liste, on peut également utiliser la fonction **remove()**
=> supprime un élément dans la liste en fonction de sa valeur

```
animaux = ["chiens", "chats", "brebis", "baleines"]
animaux.remove("chiens")
print(animaux)
```

```
['chats', 'brebis', 'baleines']
```

Syntaxe

- *Listes*

Remarque :

Lorsque l'on utilise la fonction **remove()** pour supprimer un élément qui existe plusieurs fois dans la liste, seul le premier élément est supprimé.

```
animaux = ["chiens", "chats", "brebis", "chiens"]
animaux.remove("chiens")
print(animaux)
```

```
['chats', 'brebis', 'chiens']
```

Syntaxe

- *Listes*

Opérations sur les listes :

On peut réaliser 2 types d'opérations avec les listes sur Python :

- Additionner des listes.
- Multiplier une listes.

```
animaux1 = ["chiens", "chats"]
animaux2 = ["brebis", "chiens"]
print(animaux1 + animaux2)
```

```
['chiens', 'chats', 'brebis', 'chiens']
```

```
animaux1 = ["chiens", "chats"]
print(animaux1 * 4)
```

```
['chiens', 'chats', 'chiens', 'chats', 'chiens', 'chats', 'chiens', 'chats']
```

Syntaxe

- *Listes*

Indiçage négatif :

La liste peut également être indexée avec des nombres négatifs. 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          : ["A", "B", "C", "D", "E", "F"]
#indice positif :   0    1    2    3    4    5
#indice négatif :  -6   -5   -4   -3   -2   -1
```

```
animaux = ["chiens", "chats", "brebis", "baleines"]

print(animaux[-1])
print(animaux[-2])
print(animaux[-3])
print(animaux[-4])
```

```
baleines
brebis
chats
chiens
```

Syntaxe

- *Tuples*

Un tuple est une autre **structure de données** qui de manière similaire à une liste contient une série de valeurs.
Contrairement à une liste un tuple n'est pas modifiable une fois créés, on dit qu'il n'est pas "mutable".
Un tuple est déclarée par une série de valeurs séparées par des **virgules**.

```
tableau1 = (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

tableau2 = ("err1", "err2", "err3", "err4")

print(tableau1)
```

```
(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

Remarque : Python autorise la construction de tuples contenant des valeurs de types différents.

```
tableau = ("test", 1, 2, True, 1.5)
print(tableau)
```

```
('test', 1, 2, True, 1.5)
```

Références

https://python.sdv.univ-paris-diderot.fr/04_listes/

https://python.developpez.com/cours/apprendre-python3/?page=page_11

https://www.pierre-giraud.com/python-apprendre-programmer-cours/module-paquet/

http://sosoftware.free.fr/ExosPython_CorrectionF.php

https://www.pierre-giraud.com/python-apprendre-programmer-cours/liste/

Fin

La suite :
Programmation Orienté Objet