

COURS : Tableaux en python

Les tableaux sont prévus pour stocker des valeurs dans une séquence simple (listes python), ou bien à plusieurs dimensions. Par exemple, pour stocker les notes des élèves, on peut utiliser une structure de données qui stocke les informations suivantes :

eleve	math	NSI
Adam	6	16
Julien	7	14
Anna	18	16

La première ligne a un statut différent. Il s'agit des étiquettes des données. Dans la suite de la table, chaque colonne a le même type d'information (un prenom, une note de math, ...).

On peut aussi imaginer un tableau de coordonnées temporelles, etc... Lorsque les données sont TOUTES numériques, on parle de *MATRICE*.

1.1 Liste de listes

1.1.1 Def : Il s'agit d'une liste python (mutable), où chaque élément est aussi une liste (mutable). C'est donc un objet mutable.

1.1.2 Exemple de manipulation d'un tableau

Le tableau ci-dessus peut être construit de la manière suivante :

```
1 classe_terminale = [['eleve','math','NSI','SVT'],  
2 ['Adam',6,16,10],  
3 ['Julien',7,14,11],  
4 ['Anna',18,16,17]]
```

Il est possible :

- d'atteindre l'une des lignes entière : `classe_terminale[1]` est la liste `['Adam',6,16,10]`
- de modifier un tableau à l'aide de la "notation entre crochets". Par exemple, pour ajouter 1 à la note de SVT de *Adam* : `classe_terminale[1][3] += 1`
- d'ajouter un élément en fin de tableau à l'aide de la méthode "append" : `classe_terminale.append(['Marie',16,16,16])`

Les méthodes de liste et fonctions peuvent aussi être utilisées sur un tableau.

1.1.3 Parcours par ligne ou par colonne

- Le parcours par element va permettre de traiter les lignes l'une après l'autre. On peut éliminer la première ligne pour l'itérable :

```
1 for element in classe_terminale[1:]:  
2     print(element)
```

- Le parcours par colonne va necessiter d'utiliser un indice. Supposons que l'on souhaite faire la moyenne des notes de math :

```

1 s = 0
2 for element in classe_terminale[1:]:
3     s = s + element[1]
4 moyenne_math = s / len(classe_terminale[1:])

```

1.1.4 Construction par comprehension

Le but est de simplifier le code pour le rendre plus lisible et donc plus rapide à écrire et plus simple à maintenir.

- Avec une liste simple :

```

1 new_list = [function(item) for item in list if condition(item)]

```

Exemple :

```

1 monTab1 = [let for let in 'Abracadabra' if let.upper() != 'A']
2 monTab2 = [let for let in 'Abracadabra' if let.upper() != 'A']

```

- Avec un tableau :

```

1 new_tab = [[function(j) for j in list2] for i in list1]

```

Exemple :

```

1 montab = [[let for let in 'Abracadabra'] for i in range(4)]

```

1.2 Liste de Dictionnaires

1.2.1 Def : Il s'agit d'une liste python (mutable), où chaque élément est un dictionnaire (mutable). C'est donc un objet mutable.

1.2.2 Exemple de manipulation d'un tableau

Le tableau ci-dessus peut être construit de la manière suivante :

```

1 classe_terminale = [{'eleve': 'Adam', 'math': 6, 'NSI': 16, 'SVT': 10},
2 {'eleve': 'Julien', 'math': 7, 'NSI': 14, 'SVT': 11},
3 {'eleve': 'Anna', 'math': 18, 'NSI': 16, 'SVT': 17}]

```

Il est possible :

- d'atteindre l'une des lignes entière : `classe_terminale[0]` est la liste `['eleve': 'Adam', 'math': 6, 'NSI': 16, 'SVT': 10]`
- de modifier un tableau à l'aide de la "notation entre crochets". Par exemple, pour ajouter 1 à la note de SVT de *Adam* : `classe_terminale['Adam'][3] += 1`

- d'ajouter un élément en fin de tableau à l'aide de la méthode "append" : `classe_terminale.append({'nom': 'Marie', 'math': 16, 'NSI': 16, 'SVT': 16})`

Les méthodes de liste et fonctions peuvent aussi être utilisées sur ce tableau.

1.2.3 Parcours par colonne

- Le parcours par colonne va nécessiter d'utiliser une clé. Supposons que l'on souhaite faire la moyenne des notes de math :

```
1 s = 0
2 for element in classe_terminale:
3     s = s + element['math']
4 moyenne_math = s / len(classe_terminale)
```

1.2.4 Compréhension

Pour créer un dictionnaire par compréhension :

```
1 mot = "Abracadabra"
2 lettres = ['A', 'a', 'b', 'c', 'd', 'r']
3 mondic = {lettres[i]: mot.count(lettres[i]) for i in range(len(lettres))
4           }
5 > mondic
6 {'A': 1, 'a': 4, 'b': 2, 'c': 1, 'd': 1, 'r': 2}
```

Cet exemple montre comment créer un tableau associant le nombre d'occurrence pour chaque lettre dans un mot, en utilisant la méthode count.