Tableaux à deux dimensions

Ce chapitre a pour objectif de présenter les tableaux à deux dimensions ainsi que leur implémentation et utilisation dans l'environnement Python.

I. Principe

Un tableau à deux dimensions permet de stocker un ensemble de données organisé sous forme d'une matrice sous un seul identifiant.

Dans le cas d'un tableau à une seule dimension, il est possible d'accéder à toutes les données à l'aide d'un indice. Dans le cas d'un tableau à deux dimensions, il est nécessaire d'utiliser deux indices: l'un pour la ligne et l'autre pour la colonne.

Prenons l'exemple d'un tableau à deux dimensions comprenant 4 lignes et 5 colonnes. On suppose qu'il est stocké sous le nom tab.

	0	1	2	3	4
0	7	22	31	20	9
1	2	3	5	1	8
2	23	32	19	11	48
3	51	23	41	17	121

Si on souhaite accéder à la donnée située à la quatrième ligne troisième colonne, il suffit d'utiliser d'une instruction du style tab[3][2] (bien sûr, selon le langage de programmation, la syntaxe varie). On remarquera que l'on précise la ligne avant la colonne par analogie avec les matrices mathématiques. Comme pour les tableaux à une seule dimension, il est possible de lire ou écrire les données en utilisant les indices présentés ci-dessus. Ainsi, il sera possible d'utiliser des instructions du type tab[3][2]=6 pour remplacer le 41 du tableau en 6.

II. Tableaux à deux dimensions avec Python

Python intègre, comme nous l'avons vu au chapitre précédant, les listes qui permettent de stocker des tableaux à une dimension. Les tableaux à deux dimensions utilisent, eux aussi, les listes.

Pour comprendre le principe, il faut savoir que les listes peuvent accueillir d'autres types que les types simples (entiers, flottants ou booléens). Ainsi, par exemple, une liste peut contenir des chaînes de caractères. Il est donc possible de créer une liste qui contient elle-même d'autres listes. Ainsi, la liste suivante est possible:

[1, 4, 7, 9]
[5, 3, 6]
[9, 4, 7, 6, 7]
[1, 3, 4]

Dans le cas de cet exemple, la liste contient 4 éléments qui sont, eux aussi, des listes de longueurs diverses.

Si les listes sont de même longueur, on a affaire à un tableau à deux dimensions. Ainsi, il est possible d'implémenter le tableau du premier paragraphe de la façon suivante:

[7, 22, 31, 20, 9]
[2, 3, 5, 1, 8]
[23, 32, 19, 11, 48]
[51, 23, 41, 17, 121]

Les données sont stockées à l'aide d'une liste principale composée de 4 éléments qui sont des listes de 5 éléments. Avec Python, la création de ce tableau serait la suivante:

```
tab=[[7, 22, 31, 20, 9], [2, 3, 5, 1, 8], [23, 32, 19, 11, 48], [51, 23, 41, 17, 121]]
```

Si l'on souhaite afficher la valeur stockée à la deuxième ligne troisième colonne, l'instruction est suivante:

```
>>>tab[1][2]
5
```

Note: la deuxième ligne porte le numéro 1 car on commence à 0. De même, la troisième colonne porte le numéro 2.

Il est possible d'ajouter une ligne à la fin du tableau à l'aide de l'opérateur append. Ainsi, si on souhaite ajouter une ligne contenant 1, 3, 4, 6 et 7, on pourra saisir l'instruction suivante:

```
>>>tab.append([1, 3, 4, 6, 7])
>>>tab
[[7, 22, 31, 20, 9], ….., [ 1, 3, 4, 6, 7 ]]
```

Pour supprimer une ligne, il suffit d'exploiter l'opérateur del. Par exemple, on peut utiliser l'instruction suivante:

```
>>>del tab[1]
```

Le résultat sera le suivant:

```
>>>tab
[[7, 22, 31, 20, 9], [23, 32, 19, 11, 48], [51, 23, 41, 17, 121], [ 1, 3, 4, 6, 7 ]]
```

Nous verrons, en TP, comment il est possible d'ajouter ou de supprimer une colonne en utilisant les opérateurs append et del. Ces opérations sont légèrement plus complexes.