Module 3 :: Tableau indexé

version 1.0 - dimanche 6 décembre 2020 - 17:07:50

Objectifs

☑ Connaissances visées

☐ *Tableau indexé, tableau donné en compréhension*

☑ Compétences à développer

☐ *Lire et modifier les éléments d'un tableau grâce à leurs index*
☐ *Construire un tableau par compréhension.*
☐ *Utiliser des tableaux de tableaux pour représenter des matrices : notation a [i] [j].*
☐ *Itérer sur les éléments d'un tableau.*

Ce Qu'il Faut Retenir…

Un **tableau** – nommé **liste** dans la terminologie de Python – est une **séquence ordonnée** de données.

Déclaration d'un tableau

Un jeu de n valeurs (val) peut être rangé dans un tableau de la manière suivante :

>>> t = [val_1, val_2, … ,val_n]

Les valeurs sont séparées les unes des autres par une virgule et l'ensemble des valeurs est encadré par des crochets.

Les valeurs rangées dans un tableau peuvent être de type simple : entier naturel, entier relatif, flottant, caractère, booléens ; mais aussi de type construit : chaîne de caractères, tuple, liste ou dictionnaire ;

Un tableau peut contenir des données de n'importe quel type et de types différents.

>>> t = [int_1, $char_1$, … ,str_1]

>>> t = [int_1, ($char_1$, $char_2$, … ,$char_n$), … ,[str_1, str_2, …, str_n]]

Pour créer un tableau vide il faut mettre à la suite un crochet ouvrant et un crochet fermant :

>>> t = []

Pour créer un tuple avec une seule valeur, il faut simplement mettre cette valeur entre crochets :

>>> t = [val]

La fonction 'type(t)' renvoie <class 'list'> quand une variable est de type 'tableau'

Accès à une valeur d'un tableau

Les différentes valeurs d'un tableau non vide sont accessibles par un index.

Pour accéder à une valeur, on écrit le nom du tableau suivi de son index écrit entre crochets.

>>> t[i]

Quand on lit les valeurs d'un tableau de gauche à droite, l'index de la première valeur est 'zéro'.

Quand on lit les valeurs d'un tableau de droite à gauche, l'index de la première valeur est '-1'.

Pour accéder à la valeur d'index j d'un tableau qui se trouve dans un autre tableau à l'index i, on écrit :

>>> t[i][j]

Si l'on demande à accéder à une valeur correspondant à un index qui n'existe pas – c'est à dire supérieur ou égal à 'len(t)', une exception est levée : IndexError: list index out of range

Longueur d'un tableau

La longueur d'un tableau correspond au nombre de valeurs qu'il contient.

Cette longueur est obtenue en utilisant la fonction 'len()'

>>> len(t)

Les tableaux sont concaténables

>>> t1 = [val1, val2, … , valn]

>>> t2 = [data1, data2, … , datan]

>>> t1 + t2

[val1, val2, … ,valn, data1, data2, … , datan]

Module 3 :: Tableau indexé

version 1.0 - dimanche 6 décembre 2020 - 17:07:50

Les tableaux sont duplicables

>>> t = [val_1, val_2, … , val_n]

>>> t * 2

[val_1, val_2, … ,val_n, val_1, val_2, … , val_n]

Tester la présence d'une valeur dans un tableau

Ce test s'effectue avec l'opérateur 'in' : il renvoie une valeur booléenne

>>> t = [val_1, val_2, … , val_n]

>>> val_x in t

'True' ou 'False'

Les tableaux sont itérables

>>> t = [val_1, val_2, … , val_n]

>>> for i in range(0, len(t)) :

… print(t[i])

ou

>>> for val in t :

… print(val)

Les tableaux sont mutables

Il est possible de modifier la valeur correspondant à un index existant

>>> t[1] = x

[val_1, x, … , val_n]

Il n'est pas possible d'assigner une valeur à un nouvel index i = len(t)

>>> t = [val_1, val_2, … , val_n]

>>> t[i] = val_{n+1}

IndexError: list assignment index out of range

Pour ajouter une nouvelle valeur il faut utiliser une méthode : append(), la valeur à ajouter est donnée en argument.

>>> t = [val_1, val_2, … , val_n]

>>> t.append(val_x)

[val_1, val_2, … , val_n, val_x]

La nouvelle valeur est ajoutée à la 'fin' du tableau c'est à dire à l'index i = len(t)

La méthode 'pop()' permet de lire et supprimer une valeur d'un tableau située à un index existant : 0 ≤ i < len(t). Par exemple si i vaut 1 :

>>> t = [val_1, val_2, … , val_n]

>>> t.pop(1)

val_2

t ← [val_1, val_3, … , val_n]

La longueur du tableau aura diminué d'une unité.

Pour lire et retirer la 'dernière' valeur du tableau, celle dont l'index i = len(t) – 1 , il suffit décrire :

>>> t.pop()

Il est possible de supprimer une valeur correspondant à un index existant : 0 ≤ i < len(t), en utilisant la fonction 'del()'

>>> del(t[i])

Copier le contenu d'un tableau

Soit le tableau suivant :

>>> t_1 = [val_1, val_2, … , val_n]

Pour copier le contenu de ce tableau dans un autre tableau nous sommes tenter de faire :

>>> t_2 = t_1

>>> t_2

[val_1, val_2, … , val_n]

Modifions t_1

>>> t_1.append(x)

>>> t_1

[val_1, val_2, … , val_n, x]

Mais consultons ce que contient t_2 :

>>> t_2

[val_1, val_2, … , val_n, x]

Ainsi si nous écrivons t_2 = t_1, toute modification réalisée sur t_1 affectera t_2, et inversement.

En écrivant t_2 = t_1, on ne crée pas une copie, mais un alias, c'est à dire un autre nom de variable pour le même tableau.

Pour copier un tableau il faut utiliser une méthode : 'copy()'

>>> t_2 = t_1.copy()

De cette façon toute modification réalisée sur t_1 n'affectera plus t_2, et inversement.

Matrice

Une matrice à r lignes et c colonnes est un tableau rectangulaire de r × c données qui sont toutes d'un même type, rangées ligne par ligne. Il y a r lignes, et pour chaque ligne c valeurs.

Une matrice peut être représentée par une liste de r listes. Chacune des r listes de la liste principale correspond à une ligne et possède exactement c valeurs.

Par exemple :

m = [[a,b,c],[d,e,f],[g,h,i]]

ou a, b, c, d, e, f, g, h , i sont des entiers, est une matrice de trois lignes et 3 colonnes.

Quand c = r la matrice est dite carrée.

m[x][y] permet d'accéder à la valeur d'index y de la ligne d'index x.

Module 3 :: Tableau indexé

version 1.0 - dimanche 6 décembre 2020 - 17:07:50

Tableau donné par compréhension

Il est possible de construire un tableau indexé par des instructions d'itération (boucle bornée principalement) auxquelles on peut également ajouter des instructions conditionnelles.

Par exemple : il est possible de construire une liste des 20 premiers entiers naturels pairs de la façon suivante :

```
# construction d'une liste par instruction itérative
t1i=[]
for i in range (2,21,2):
     t1i.append(i)
# construction de la même liste par compréhension
t1c = [x for x in range(2,21,2)]
```

Les compréhensions de listes fournissent un moyen de construire des listes de manière très concise.

Une compréhension de liste consiste à placer entre crochets une expression suivie par une clause `for` puis par zéro ou plus clauses `for` ou `if`. Le résultat est une nouvelle liste résultat de l'évaluation de l'expression dans le contexte des clauses `for` et `if` qui la suivent.

```
# construction d'une liste par instructions itérative et conditionnelle
t2ic=[]
for x in range (0, 1000):
    if x % 3 == 0:
        t2ic.append(x)
# construction de la même liste par compréhension
t2co=[x for x in range(0,1000) if x % 3 == 0]
```

Il est possible de construire par compréhension une liste de tuples, ou une liste de listes.

```
# construction d'une liste de tuples par compréhension
t3=[(x, 1/x) for x in range(1,81)]


# construction d'une liste de listes par compréhension
t4=[ [x,x/2, (x+3)/2] for x in (2, 5, 8, 13, 15)]
```

Autres exemples de constructions possibles :

```
x=[1,2,3]
y=[4,5,6]
z1=[ [ x[i],y[i] ] for i in range(0,3) ]
z2=[ (i,j) for i in x for j in y]
```

Pour aller plus loin :

https://docs.python.org/fr/3/tutorial/datastructures.html#more-on-lists
https://docs.python.org/fr/3/tutorial/datastructures.html#list-comprehensions
https://docs.python.org/fr/3/tutorial/datastructures.html#nested-list-comprehensions