

LINGE1225 : Programmation en économie et gestion

Cours 5

Listes imbriquées et tableaux

François Fouss & Marco Saerens

Année académique 2019-2020

Livre de référence

- Chapitre 10 : Approfondir les structures de données



Plan

1. Listes (rappel)
2. Matrice
 - liste à deux dimensions : façon compacte de construire une matrice
 - Tableaux
3. Listes multidimensionnelles
4. Application financière

3

Listes

RAPPEL

- Les *listes* sont des collections ordonnées d'objets.
- Les listes sont délimitées à l'aide de crochets :

#listes

```
nombres = [5, 38, 10, 25]
mots = ["jambon", "fromage", "confiture", "chocolat"]
Stuff = [5000, "Brigitte", 3.1416, ["Alberté", "René", 1947]]
```

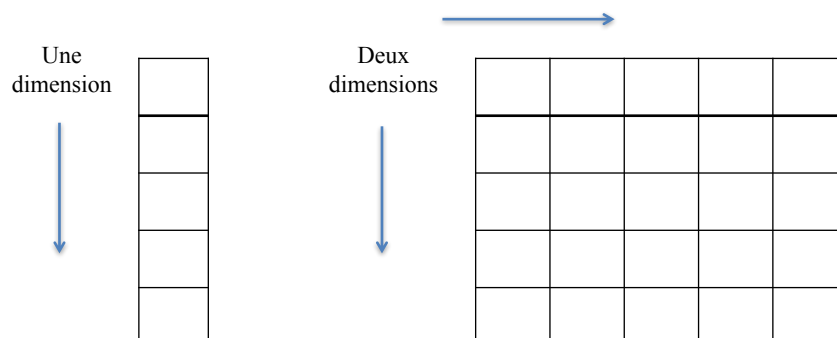
- Dans une liste, il est possible de combiner des données de n'importe quel type, y compris des listes et des dictionnaires.

4

Matrices

Listes imbriquées

- Une *liste à une dimension* stocke une liste d'éléments.
- Une *liste à deux dimensions* peut être vue comme une matrice (avec des colonnes et des lignes).



Remarque : Une liste à une dimension peut être vue comme un vecteur.

6

Création d'une liste à deux dimensions

- Une liste à deux dimensions peut être déclarée à l'aide d'une double boucle `for` précisant le nombre de colonnes et de lignes.

```
mat=[0 for j in range(5)] for i in range(4)]
```

Equivalent:

```
mat = []
for i in range(4):
    row = []
    mat.append (row)
    for j in range(5):
        row.append(0)
```

- La première boucle gère les colonnes.
- La deuxième boucle gère les lignes.

- Cela va créer une matrice de 5 colonnes et 4 lignes.

	0	1	2	3	4
0					
1					
2					
3					

7

Créer une liste imbriquée avec une boucle *for* et la multiplication *

- Créer une matrice à l'aide d'une boucle `for`.

Matrice de n sur m remplie de zéro :

```
n = 3
m = 4
a = [0] * n
for i in range(n):
    a[i] = [0] * m
```

création d'une liste avec 3 éléments

répéter 0 quatre fois
pour chaque ligne

- Résultat :

```
[[0, 0, 0, 0], [0, 0, 0, 0], [0, 0, 0, 0]]
```

Résultat matriciel :

0	0	0	0
0	0	0	0
0	0	0	0

8

Listes imbriquées : Enumerate

- Certaines fonctions en Python créent des listes imbriquées

Exemple : `enumerate(1)`

```
#listes imbriquées
```

```
l = [ "Bruxelles", "Mons", "Anvers" ]
for t in enumerate(l):
    print(t)
```

```
(0, "Bruxelles")
(1, "Mons")
(2, "Anvers")
```

À titre d'information, les résultats sont des « tuples ». Les tuples sont un type de données que nous ne verrons pas dans le cadre de ce cours. Voici un lien si vous voulez en savoir plus : <https://python.doctor/page-apprendre-tuples-tuple-python>

9

Extraire une valeur d'une matrice (liste à deux dimensions)

- Pour récupérer la valeur contenue dans un élément d'une matrice, on spécifie le nom de la variable qui contient cette matrice, le numéro de ligne de l'élément suivi du numéro de colonne de l'élément.
- Je veux récupérer le numéro 9 dans la matrice.

	0	1	2	3	4
0					
1		9			
2					
3					

- Commande :

```
print(mat[1][1])
```

9

Exemple en Python :

```
lst = [[1,2,3],[4,5,6]]
print (lst[1][2])
6
```

10

Matrice

- Une liste à deux dimensions est une liste de listes qui peut être vue comme un tableau ou une matrice.
- Exemple : Matrice à deux dimensions.

```
#matrice 2D
a=[[1,2], [3,4], [5,6]]
```

Création d'une matrice trois lignes deux colonnes

```
a
[[1,2], [3,4], [5,6]]
```

- Résultat matriciel :

1	2
3	4
5	6

11

Matrice

- Chaque liste a une longueur constante.
- On peut donc récupérer la longueur des colonnes mais aussi des lignes (= le nombre de colonnes et de lignes) avec la fonction *len()*.

```
#matrice 2D
a=[[1,2], [3,4], [5,6]]

len(a) ← nombre de lignes = taille de la liste
3

len(a[1]) ← Nombre des colonnes= taille d'une liste imbriquée
2
```

12

Matrice

- Un tableau de 2 dimensions (liste de listes) contiendra des éléments qui devront être appelés par la double indexation `[i][j]`
- Le premier index renvoie donc à l'index de la ligne
- Le deuxième index renvoie à l'index de la colonne
- La commande `len()` renvoie la longueur de la liste

```
#matrice 2D
a[1] ← Appel de la deuxième ligne
[3, 4]

a[1][0] ← Appel de l'élément deuxième ligne, première colonne
3

len(a) ← Longueur de la liste 1D
3

len(a[1]) ← Longueur de la liste 2D
2
```

13

Matrice

Expression	Type	Description
<code>scores</code>	<code>int</code>	Nombre entier.
<code>scores[5]</code>	<code>int[]</code>	Liste de nombres entiers.
<code>scores[5][12]</code>	<code>int[][]</code>	Liste à deux dimensions de nombres entiers.

14

Listes imbriquées : Exercice

Créez une matrice de 4 sur 5 remplie de nombres entiers incrémentés de 1. Commencez à 100.

Vous devriez obtenir quelque chose comme cela:

```
100 101 102 103 104
105 106 107 108 109
110 111 112 113 114
115 116 117 118 119
```

15

Solution

```
a=[[0 for j in range(5)] for i in range(4)]
i=100
for j in range(len(a)):
    for k in range(len(a[0])):
        a[j][k]=i
        i+=1
```

Renvoie le nombre
de colonnes

```
100 101 102 103 104
105 106 107 108 109
110 111 112 113 114
115 116 117 118 119
```

Remarque : on suppose que toutes les lignes sont de la même taille.

16

Tableau

- Pour afficher un tableau à deux dimensions, il est possible d'utiliser des boucles imbriquées.

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

```
for i in range(len(a)):  
    for j in range(len(a[i])):  
        print(a[i][j], end=" ")  
    print()
```

```
1 2 3  
3 4 5
```

La première
boucle parcourt les
lignes

La deuxième
boucle parcourt les
colonnes/rangées.

= afficher la liste ligne par ligne,
les nombres étant séparés par des
espaces.

On imprime
l'élément se
trouvant à la ligne
i et à la colonne j

17

Tableau : parcours de la boucle

	j=0	j=1	j=2	
i=0	1	2	3	matrice a
i=1	3	4	5	

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

```
a= [[1,2,3], [3,4,5]]  
for i in range(len(a)):  
    for j in range(len(a[i])):  
        print(a[i][j], end=" ")  
    print()
```

Pour chaque ligne

i=0
j=0 → 1
j=1 → 2
j=2 → 3

i=1
j=0 → 3
j=1 → 4
j=2 → 5

Imprime l'élément
correspondant de la colonne

18


Listes multidimensionnelles

Liste multidimensionnelle

- Une liste peut avoir plusieurs dimensions.
- Une liste à plus d'une dimension est appelée liste multidimensionnelle.
- Chaque liste a une longueur définie (`len()`).
- Parce que chaque dimension est une référence de liste de listes, les listes dans une dimension peuvent être de différentes longueurs.

Liste multidimensionnelle – exemple

```
lst = [[1, 6, [8, 3], 3], [9, [8, [[7, 6], 1]]]]
```



```
lst[1][1][1][0][0] # donne 7
```

21

Application financière



Application 5

Vous cherchez à faire un prêt à la banque. Vous avez donc été récolter les informations quant aux taux pour chaque banque. Créez un tableau en Python qui permet de comparer toutes les banques selon le capital acquis étant donné que :

AXXA = 2,3% (2 ans)
 BELFIOUS = 1,03% (1 an)
 INJ = 0,5% (6 mois)
 CBK = 2,1% (1 an)

Les taux ont été donnés pour différentes maturités. Comparez donc les banques suivant le capital acquis après 6 ans selon la méthode de l'intérêt simple. Vous comptez placer 5000 €.

23

Solution

```
tablebanque = [[0 for i in range(2)] for j in range(4)]
tablebanque

banque = ["AXXA", "BELFIOUS", "INJ", "CBK"]

c0 = 5000
c_AXXA = (0.023 * 3 * c0) + c0
c_Belfious = (0.0103 * 6 * c0) + c0
c_INJ = (0.005 * 12 * c0) + c0
c_CBK = (0.021 * 6 * c0) + c0

capital_acq = [c_AXXA, c_Belfious, c_INJ, c_CBK]
```

24

Solution (suite)

➤ Affichage du tableau

```
for i in range(len(tablebanque)):
    tablebanque[i][0] = banque[i]
    tablebanque[i][1] = capital_acq[i]

tablebanque

for i in range(len(tablebanque)):
    for j in range(len(tablebanque[i])):
        print(tablebanque[i][j], end = " ")
    print()
```

Résultat :

```
AXXA 5345.0
BELFIUS 5309.0
INJ 5300.0
CBK 5630.0
```

25