Les Structures de Données Python
Cours 4 : Les listes multidimensionnelles

Halim Djerroud

ESIEE
PARIS

révision : 0.1

Plan

Les listes 2D :

1. Définition
2. Déclaration et initialisation
3. Parcourir une liste multidimensionnelle

Représentation générale des listes 2D

- Une liste classique contient de plusieurs éléments
- **Et si ces éléments son encore des listes ?**
  - Résultat : une liste multidimensionnelle
  - Dans ce cas on parle de tableau multidimensionnel, ayant plusieurs lignes et plusieurs colonnes
  - Le plus courant est le tableau à 2 dimensions

Exemple :

- Grille de pixels d'un écran
- Matrices pour calcul mathématique

Représentation en Python

- Une liste à 2D (matrice) = Une liste 1D où chacune de ses valeurs est elle-même une liste 1D séparées par des virgules

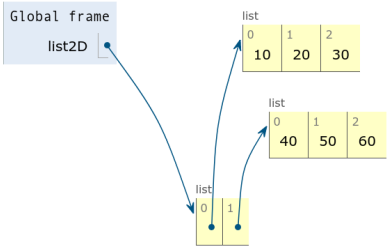
```
Liste2D = [[lst_1] , [lst_2] , ... ,  [lst_n] ]
```

Pour accéder à une valeur

Pour accéder à une valeur stockée dans une liste 2D, il faut :

- D'abord accéder à la ligne par le premier indice
- Puis en utilisant un deuxième indice, accéder à une colonne

```
liste2D = [[10, 20, 30],
           [40, 50, 60]
          ]
print(liste2D[0][1])   # => 20
print(liste2D[1][1])   # => 50
```



Création d'une liste

- Quelques créations de listes 2D pré-remplies
- La fonction `print()` permet d'afficher les valeurs d'une liste2D

```
# Creation et initialisation d'une liste 2D
# nombre de ligne = nombre de colonnes = 3
liste2D = [[1, 2, 3],
           [4, 5, 6],
           [7, 8, 9]
          ]
print(liste2D)
# => [[1, 2, 3],[4, 5, 6],[7, 8, 9]]
```

Initialiser une liste 2D

- Initialiser une liste 2D avec une valeur unique

```
col, lin = 3, 5
liste2D = []
for i in range(col):
    line = []
    for j in range(lin):
        line.append(0)
    liste2D.append(line)

print (liste2D)
```

Initialiser une liste 2D

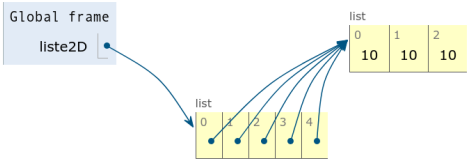
- Initialiser une liste 2D avec une valeur unique

```
col, lin = 3, 5
liste2D = [[0 for i in range(lin)] for j in range(col)]

# => [[0, 0, 0, 0, 0],
#     [0, 0, 0, 0, 0],
#     [0, 0, 0, 0, 0]]
```

Initialiser une liste 2D

- Méthode `[x] * N` pour initier un tableau 2D

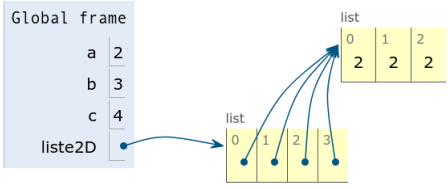
```
liste2D =  [[10] * 3] * 5
# =>[[10, 10, 10],
#    [10, 10, 10],
#    [10, 10, 10],
#    [10, 10, 10],
#    [10, 10, 10]]
```



Initialiser une liste 2D

Exemple :

```
a, b, c = 2, 3, 4
liste2D =  [[a] * b] * c

# => [[2, 2, 2],
#     [2, 2, 2],
#     [2, 2, 2],
#     [2, 2, 2]]
```



Parcourir une liste 2D

Via les indices

- Deux boucles imbriquées
  - Une boucle externe qui va parcourir les lignes (taille de la liste 2D)
  - Une boucle interne qui va parcourir les colonnes (taille de chaque valeur de la liste 2D)

```
# Initialisation d'une liste 2D
liste2D = [[1, 2, 3], [4, 5, 6]]
# Parcourir les valeurs de la liste (valeurs = nombre de lignes)
for i in range(len(liste2D)):
    # La taille de chaque ligne represente le nombre de colonnes
    for j in range(len(liste2D[i])):
        # Affichage des valeurs de la ligne i
        print(liste2D[i][j], end=' ')
    # retour a la ligne avant la prochaine ligne
    print()
```

Parcourir une liste 2D

Via les itérateurs

- Deux boucles imbriquées
  - Une boucle externe qui va parcourir les lignes (chacune des valeurs de la liste 2D)
  - Une boucle interne qui va parcourir les colonnes (Les valeurs de la liste sur une case)

```
# Initialisation d'une liste 2D
liste2D = [[1, 2, 3], [4, 5, 6]]
# Parcourir chaque case (une case contient une liste)
for row in liste2D:
    for elem in row:
        print(elem, end=' ')
    print()
```

Remplir une liste 2D

Exemple de remplissage par des données saisies par l'utilisateur

```
# Initialisation d'une liste 2D
liste2D = []
nb_lig = int(input("Saisir un nombre de lignes : "))
nb_col = int(input("Saisir un nombre de colonnes : "))
for i in range(nb_lig):
    # declarer une liste 1D vide qui sera remplie par l'utilisateur
    L=[]
    for i in range(nb_col):
        val = int(input("Saisir une valeur : "))
        # Ajouter la valeur saisie a la liste L
        L.append(val)
    # une fois nb_col valeurs sont saisies; sont ajoutees a liste2D
    # Boucler pour saisir la prochaine ligne
    liste2D.append(L)
    print(liste2D
```

liste 2D : Exemple 1

```
M = [
      [0  , -1 , 10],
      [5  , -7 ,  6],
      [22 , 0  , 16],
      [9  , 31 ,  8]
    ]
```

- Écrire un programme Python qui permet d'afficher le contenu des colonnes d'indices impairs de la matrice (liste 2D) M ci-dessous sous la forme :

```
-1
-7
0
31
```

Listes 2D : Exemple 1

```
M = [[0, -1, 10], [5, -7, 6], [22, 0, 16], [9, 31, 8]]
tmp = []
nb_lig = len(M)
nb_col = len(M[0])

for i in range(nb_col):
    L=[]
    for j in range(nb_lig):
        L.append(M[j][i])
    tmp.append(L)

for r in range(len(tmp)):
    for k in range(len(tmp[0])):
        if r % 2 == 1:
            print(tmp[r][k])
```

Listes 2D : Exemple 2

Écrire un programme Python qui :

- demande à l'utilisateur de saisir le nombre de lignes nbLig
- demande à l'utilisateur de saisir le nombre de colonnes nbCol
- remplit une matrice (liste 2D) M de taille nbLig x nbCol
- crée une liste 1D L dans laquelle il stocke les sommes de chacune des colonnes de M
- affiche la liste L (affichage standard)

liste 2D : Exemple 2

```
# Initialisation d'une liste 2D
M = []
nb_lig= int(input("Saisir le nombre de lignes : "))
nb_col= int(input("Saisir le nombre de colonnes  : "))

for i in range(nb_lig):
    L=[]
    for i in range(nb_col):
        val = int(input("Enter a value : "))
        L.append(val)
    M.append(L)

L = [0] * nb_col
for i in range(nb_lig):
    for j in range(nb_col) :
        L[j] += M[i][j]
print(L)
```

Listes 2D : Exemple 3

- Écrire un programme Python qui permet d'afficher le contenu de la matrice ci-dessous dans l'ordre inverse.

```
M = [
      [0  , -1 , 10],
      [5  , -7 ,  6],
      [22 , 0  , 16],
      [9  , 31 ,  8]
    ]
>> 8, 31, 9, 16, ...
```

Listes 2D : Exemple 3

```
nbLig = len(M)
nbCol = len(M[0])

for i in range(nbLig-1, -1, -1):
    for j in range (nbCol-1, -1, -1):
        print(M[i][j])
```

Listes 2D : Exemple 3

Écrire un programme Python qui :

- remplit une matrice (liste 2D) M1 de taille nbLig x nbCol
- remplit une matrice (liste 2D) M2 de même taille nbLig x nbCol
- calcule la somme de M1 et M2
- affiche à l'écran la matrice résultante

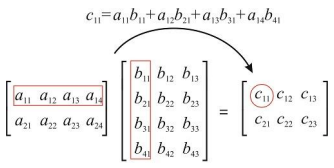
Listes 2D : Exemple 3

```
nbLig = int(input("Saisir le nombre de lignes : "))
nbCol = int(input("Saisir le nombre de colonnes : "))

M1, M2, M3 = [], [], []

for i in range(nbLig):
    temp = []
    for j in range(nbCol):
        temp.append(int(input("Saisir une valeur : ")))
    M1.append(temp)

for i in range(nbLig):
    temp = []
    for j in range(nbCol):
        temp.append(int(input("Saisir une valeur : ")))
    M2.append(temp)

for i in range(nbLig):
    temp = []
    for j in range(nbCol):
        temp.append(M1[i][j] + M2[i][j])
    M3.append(temp)

print(M3)
```

Listes 2D : Exemple 3, produit matriciel

Écrire un programme Python qui :

- remplit une matrice (liste 2D) M1 de taille nbLig1 x nbCol1
- remplit une matrice (liste 2D) M2 de taille nbLig2 x nbCol2 avec nbCol1 = nbLig2
- alcule le produit de M1 et M2
- affiche à l'écran la matrice résultante sous la forme mathématique



Listes 2D : Exemple 3, produit matriciel

```
M1, M2, P = [], [], []

nb_lig1= int(input("Saisir le nombre de lignes: "))
nb_col1= int(input("Saisir le nombre de colonnes : "))

print ("==== Filling M1 ====")
for i in range(nb_lig1):
    L=[]
    for i in range(nb_col1):
        val = int(input("sasir une valeur : "))
        L.append(val)
    M1.append(L)

nb_lig2 = int(input("Saisir le nombre de lignes: "))
while nb_lig2 <= 0 or nb_lig2 != nb_col1:
    nb_lig2= int(input("Produit matriciel impossible, Saisir le nombre de lignes :"))

nb_col2 = -1
while nb_col2 <= 0:
    nb_col2= int(input("Saisir le nombre de colonnes : "))
```

Listes 2D : Exemple 3, produit matriciel

```
for i in range(nb_lig2):
    L=[]
    for i in range(nb_col2):
        val = int(input("sasir une valeur : "))
        L.append(val)
    M2.append(L)

for i in range(len(M1)):
   L=[]
   for j in range(len(M2[0])):
       result = 0
       for k in range(len(M2)):
           result += M1[i][k] * M2[k][j]
       L.append(result)
   P.append(L)

for row in P:
    for val in row:
        print(val, end="\t")
    print()
```

Listes 2D

- Écrire un programme Python qui :
  - remplit une matrice (liste 2D) M de taille nbLig x nbCol
  - calcule la transposée de la matrice M
  - affiche à l'écran la matrice résultante sous la forme mathématique