

# Programmation PYTHON

Cours 7

Nassim ZELLAL

2020/2021

# Structures de données multidimensionnelles

- Manipuler des listes multidimensionnelles.
- Manipuler des dictionnaires multidimensionnels.
- Des listes de listes.
- Des dictionnaires de dictionnaires.

## Liste de listes (matrice)

- Stocker les valeurs suivantes dans une liste à deux dimensions (ou bidimensionnelle). Dans cette liste, il y a 3 lignes de 3 valeurs par ligne (structure de données bidimensionnelle) :
- Exemple d'une matrice 3x3 (3 lignes et 3 colonnes).
- **15,20,25**
- **30,35,40**
- **70,80,90**

# Alimentation et affichage d'une liste à deux dimensions

- Prenons l'exemple de la liste **M** (matrice du précédent slide), qui contient donc 3 listes.
- **M**=[[15,20,25],
- [30,35,40],
- [70,80,90]
- ]
- `print(M[1][2])`
- `print("-----")`
- `print("Nombre de lignes",len(M))`
- `print("-----")`
- `print("Nombre de colonnes",len(M[0]))`
- `print("-----")`
- `for i in M:`
- `print("Élément de la première colonne :",i[0])`

## Alimentation et affichage d'une liste à deux dimensions

- `ligne1 = [15,20,25]`
- `ligne2 = [30,35,40]`
- `ligne3 = [70,80,90]`
- `M = [ligne1,ligne2,ligne3]`
- `print(type(M))`
- `print(M[1][2])`
- `print("-----")`
- `print("Nombre de lignes",len(M))`
- `print("-----")`
- `print("Nombre de colonnes",len(M[0]))`
- `print("-----")`
- `for i in M:`
  - `print("Élément de la première colonne :",i[0])`

# Alimentation et affichage d'une liste à deux dimensions

```
C:\Users\user\Desktop>test.py
<class 'list'>
40
-----
Nombre de lignes 3
-----
Nombre de colonnes 3
-----
Élément de la première colonne : 15
Élément de la première colonne : 30
Élément de la première colonne : 70
```

## Alimentation et affichage d'une liste à deux dimensions

- `M=[[15,20,25],[30,35,40],[70,80,90]]`
- `for a in M:`
- `print("Boucle for 1",a)`
- `for b in a:`
- `print("Boucle for 2",b)`

## Alimentation et affichage d'une liste à deux dimensions

```
C:\Users\user\Desktop>test.py
Boucle for 1 [15, 20, 25]
Boucle for 2 15
Boucle for 2 20
Boucle for 2 25
Boucle for 1 [30, 35, 40]
Boucle for 2 30
Boucle for 2 35
Boucle for 2 40
Boucle for 1 [70, 80, 90]
Boucle for 2 70
Boucle for 2 80
Boucle for 2 90
```



# Alimentation et affichage d'une liste à deux dimensions

- `M=[[15,20,25],[30,"orange",40],[70,80,90]]`
- `M[1][1]="pomme"`
- `for a in M:`
  - `print("Boucle for 1",a)`
  - `for b in a:`
    - `print("Boucle for 2",b)`

## Alimentation et affichage d'une liste à deux dimensions

```
Boucle for 1 [15, 20, 25]
Boucle for 2 15
Boucle for 2 20
Boucle for 2 25
Boucle for 1 [30, 'pomme', 40]
Boucle for 2 30
Boucle for 2 pomme
Boucle for 2 40
Boucle for 1 [70, 80, 90]
Boucle for 2 70
Boucle for 2 80
Boucle for 2 90
```

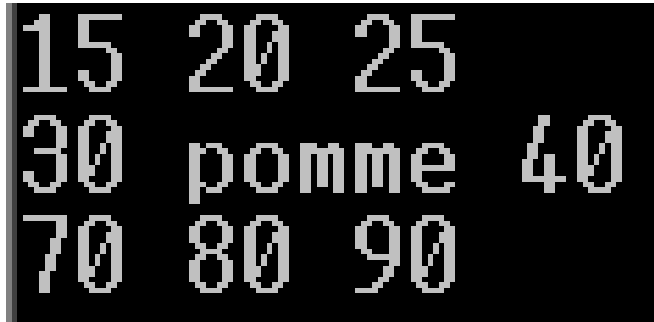
# Alimentation et affichage d'une liste à deux dimensions

- `import re`
- `M=[[15,20,25],[30,"orange",40],[70,80,90]]`
- `M[1][1]="pomme"`
- `for a in M:`
- `print(re.sub("[',\[\]]", "", str(a)))`
- `#-----#`
- `M=[[15,20,25],[30,"orange",40],[70,80,90]]`
- `M[1][1]="pomme"`
- `for a in M:`
- `print(str(a).replace("'", "").replace(", ", "").replace("[", "").replace("]", ""))`

## Alimentation et affichage d'une liste à deux dimensions

- `M=[[15,20,25],[30,"orange",40],[70,80,90]]`
- `M[1][1]="pomme"`
- `for i in range(len(M)):`
  - `for j in range(len(M[i])):`
    - `print(M[i][j], end=' ')`
  - `print()`

# Alimentation et affichage d'une liste à deux dimensions



A 3x3 grid of numbers and text displayed on a black background. The grid contains the following values:

15	20	25
30	pomme	40
70	80	90

## Alimentation et affichage d'une liste à deux dimensions - append()

- `M=[]`
- `a=[15,20,25]`
- `b=[30,35,40]`
- `c=[70,80,90]`
- `M.append(a)`
- `M.append(b)`
- `M.append(c)`
- `print(M)`
- `> [[15, 20, 25], [30, 35, 40], [70, 80, 90]]`

## Alimentation et affichage d'une liste à deux dimensions - append()

- `M=[]`
- `a=[15,20,25]`
- `b=[30,35,40]`
- `c=[70,80,90]`
- `M.append(a)`
- `M.append(b)`
- `M.append(c)`
- `for a in M:`
  - `print("Boucle for 1",a)`
  - `for b in a:`
    - `print("Boucle for 2",b)`

## Alimentation et affichage d'une liste à deux dimensions - append()

```
C:\Users\user\Desktop>test.py
Boucle for 1 [15, 20, 25]
Boucle for 2 15
Boucle for 2 20
Boucle for 2 25
Boucle for 1 [30, 35, 40]
Boucle for 2 30
Boucle for 2 35
Boucle for 2 40
Boucle for 1 [70, 80, 90]
Boucle for 2 70
Boucle for 2 80
Boucle for 2 90
```



# Alimentation et affichage d'un dictionnaire de dictionnaires

- `dic={1:{1:15,2:20,3:25},`
- `2:{1:30,2:35,4:40},`
- `3:{7:70,6:80,8:90}`
- `}`
- `print(dic[3][6])`
- `>80`
- `print(dic[3].get(6))`
- `>80`

# Alimentation et affichage d'un dictionnaire de dictionnaires

- `dic={1:{1:15,2:20,3:25},`
- `2:{1:30,2:35,4:40},`
- `3:{7:70,6:80,8:90}`
- `}`
- `for a in dic:`
- `print("clé ==> ",a,"valeur ==>",dic.get(a))`

# Alimentation et affichage d'un dictionnaire de dictionnaires

```
C:\Users\user\Desktop>test.py  
clé ==> 1 valeur ==> {1: 15, 2: 20, 3: 25}  
clé ==> 2 valeur ==> {1: 30, 2: 35, 4: 40}  
clé ==> 3 valeur ==> {8: 90, 6: 80, 7: 70}
```

## Alimentation et affichage d'un dictionnaire de dictionnaires

- `dic={1:{1:15,2:20,3:25},`
- `2:{1:30,2:35,4:40},`
- `3:{7:70,6:80,8:90}`
- `}`
- `for a in dic:`
- `for b in dic.get(a):`
- `print("clé ==> ",b,"valeur ==> ",dic[a][b])`
- `#ou bien print("clé ==> ",b,"valeur ==> ",dic.get(a).get(b))`

# Alimentation et affichage d'un dictionnaire de dictionnaires

```
clé ==> 1 valeur ==> 15  
clé ==> 2 valeur ==> 20  
clé ==> 3 valeur ==> 25  
clé ==> 1 valeur ==> 30  
clé ==> 2 valeur ==> 35  
clé ==> 4 valeur ==> 40  
clé ==> 7 valeur ==> 70  
clé ==> 6 valeur ==> 80  
clé ==> 8 valeur ==> 90
```

## Exercice 1

- Construire un dictionnaire ayant comme clé la chaîne de caractères "arbre" et sa valeur un dictionnaire ayant comme clé "feuille" et sa valeur la liste [3,5,2]. Et une autre clé "ar" et sa valeur un dictionnaire ayant comme clé "feu" et sa valeur la liste [99,8,2]. Et enfin, une autre clé "a" et sa valeur un dictionnaire ayant comme clé "fe" et sa valeur la liste [999,8,2].
- Afficher tout le contenu de cette structure de données.

---

## Exercice 2

- **Construire automatiquement une liste contenant trois listes, chacune contenant les entiers : 0,1,2,3 trois fois.**
  - **Utiliser la méthode « `append()` ».**
-

## Exercice 3

- Faire un script qui prend en entrée le fichier « verbes.txt ». Ce dernier est un fichier TSV (Tab-separated values) encodé en UTF-8. Il est composé de 5 colonnes (lemme vocalisé, racine, mode de conjugaison (6 types), transitivité, identifiant).
- Le script doit regrouper dans une sortie XML chaque racine avec son/ses lemme(s), le mode de conjugaison (mode) et la transitivité. La transitivité doit être déduite de « verbes.txt » (م => T, ل=>I, ك=>IT).
- La sortie doit être triée selon la racine du verbe par ordre croissant, en respectant le format suivant :
  - **<Verbes>**
  - **<Verbe racine="xxx">**
  - **<Info mode="xxx" lemme=" xxx " transitivité="T"/>**
  - **</Verbe>**
  - **<Verbe racine="xxx">**
  - **<Info mode="xxx" lemme=" xxx " transitivité="I"/>**
  - **<Info mode="xxx" lemme=" xxx " transitivité="IT"/>**
  - **</Verbe>**
  - **</Verbes>**



## Exercice 3 - verbes.txt → res.xml

#	الفعل	الجزر	T	باب التصريف	id
1	أَبَا	3	م	ت1	
2	أَبَا	1	ك	ت2	
3	أَبَا	2	ك	ت3	
4	أَبَا	1	ل	ت4	
5	أَبَا	2	ل	ت5	
6	أَبَا	4	ل	ت6	
7	أَبَا	2	ك	ت7	
8	أَبَا	4	ل	ت8	
9	أَبَا	1	ل	ت9	
10	أَبَا	2	ل	ت10	
11	أَبَا	4	ل	ت11	
12	أَبَا	1	ك	ت12	
13	أَبَا	2	ك	ت13	
14	أَبَا	4	ل	ت14	

```

1 <Verbes>
2   <Verbe racine="أب">
3     <Info mode="3" lemme="أَبَا" transitivité="T"/>
4   </Verbe>
5   <Verbe racine="أب">
6     <Info mode="1" lemme="أَب" transitivité="IT"/>
7     <Info mode="2" lemme="أَب" transitivité="IT"/>
8   </Verbe>
9   <Verbe racine="أب">
10    <Info mode="1" lemme="أَبْت" transitivité="I"/>
11    <Info mode="2" lemme="أَبْت" transitivité="I"/>
12    <Info mode="4" lemme="أَيْت" transitivité="I"/>
13  </Verbe>
14  <Verbe racine="أب">
15    <Info mode="2" lemme="أَبْث" transitivité="IT"/>
16    <Info mode="4" lemme="أَيْث" transitivité="I"/>
17  </Verbe>
18  <Verbe racine="أب">
19    <Info mode="1" lemme="أَبْد" transitivité="I"/>
20    <Info mode="2" lemme="أَبْد" transitivité="I"/>
21    <Info mode="4" lemme="أَيْد" transitivité="I"/>
22  </Verbe>
23  <Verbe racine="أب">
24    <Info mode="1" lemme="أَبْر" transitivité="IT"/>
25    <Info mode="2" lemme="أَبْر" transitivité="IT"/>
26    <Info mode="4" lemme="أَيْر" transitivité="I"/>
27  </Verbe>
28 </Verbes>

```