

## A. Tuples ou p-uplets :

### Ce qu'il faut savoir :

- **Un p-uplet** est une suite ordonnée d'éléments (nombres, caractères, booléens, autres p-uplets...) qui peuvent être de n'importe quel type de données. On les appelle aussi **des tuples**.
- Pour créer un **p-uplet** nommé « **t** », il faut écrire **p** valeurs séparées par des virgules et entourée ou non par des parenthèses.

**Exemples :**

- $t=12,23,10,46,9$  ou  $t=(12,23,10,46,9)$  représentent le même 5-uplet.
- $t="a","b","c","d"$  ou  $t=("a","b","c","d")$  représentent le même 4-uplet.
- $t=True,5,"e"$  ou  $t=(True,5,"e")$  représentent le même 3-uplet.
- $t=12,(23,10,46,9)$  ou  $t=(12,(23,10,46,9))$  représentent le même 2-uplet.
- $t=()$  qui est **le tuple vide** ou 0-uplet (sans aucun élément).

- On peut faire **une concaténation** deux p-uplets avec l'opérateur « **+** » ou l'opérateur « **\*** ».

**Exemples :**

- si  $t_1=("a","b")$  et  $t_2=("c","d")$ , alors  $t_1+t_2$  vaut  $("a","b","c","d")$ .
- si  $t_3=(3.5,"alain")$  alors  $3 * t_3=(3.5,"alain",3.5,"alain",3.5,"alain")$

- Pour vérifier l'appartenance d'un élément à un p-uplet, on peut utiliser le mot « **in** ».

**Exemples :**

- si  $t=(4,6,8,10)$ , alors :  $8 \text{ in } t$  donne `True` et  $5 \text{ in } t$  donne `False`.

- **La taille d'un tuple** s'obtient avec la méthode « **len** ».

**Exemples :** Si  $t_1=(2,3,4,5)$ ,  $t_2=("a",True)$  et  $t_3=()$  alors  $\text{len}(t_1)$ ,  $\text{len}(t_2)$  et  $\text{len}(t_3)$  valent 4, 2 et 0.

- **Les indices** permettent d'obtenir les éléments d'un p-uplets.

Ainsi, si  $t$  est une liste et si  $p=\text{len}(t)$ , alors  $t=(t[0],t[1],t[2],...,t[p-2],t[p-1])$ .

**Exemples :** Si  $t=(2, False, "nsi", (x,y))$ . Alors  $\text{len}(t)$  vaut 4

- $t[0]$  vaut 2.  $t[2]$  vaut "nsi".  $t[3]$  vaut  $(x,y)$ .  $t[-1]$  vaut  $(x,y)$ .  $t[-3]$  vaut `False`.
- $t[1:3]$  affichera  $t[1]$  et  $t[2]$  (pas  $t[3]$ !), à savoir  $(False,"nsi")$ .
- $t[2:4]$  affichera  $t[2]$  et  $t[3]$  (pas  $t[4]$ !), à savoir  $( "nsi" , (x,y) )$ .

- **Remarque :** Les éléments d'un tuple existant ne sont pas modifiables par une affectation du type  $t[i]=...$ . Une telle instruction renvoie une erreur.

**Exercice 1 :** Pour chaque cas, détermine ce que va afficher Python (après avoir répondu, corrige toi en tapant les instructions sur la console de Python).

1.

```
>>> t1=("vert","rouge","jaune")
>>> t2=(1,2,3)
>>> t1+t2
```

2.

```
>>> t1=("vert","rouge","jaune")
>>> t2=(1,2,3)
>>> 2*t2
```

3.

```
>>> t1=("vert","rouge","jaune")
>>> t2=(1,2,3)
>>> 2*t1+t2
```

4.

```
>>> a,b=3,6
>>> a,b=b,a
>>> (a,b)
```

5.

```
>>> l=("jean","alain","marc")
>>> len(l)
```

6.

```
>>> l=("jean","alain","marc")
>>> len(5*l)
```

7.

```
>>> l=("ali","DIOUF",30,"DAKAR")
>>> l[2]
```

8.

```
>>> l=("ali","DIOUF",30,"DAKAR")
>>> l[1:3]
```

9.

```
>>> l=("ali","DIOUF",30,"DAKAR")
>>> l[-3]
```

10.

```
>>> l=("ali","DIOUF",30,"DAKAR")
>>> l[3]="PARIS"
```

11.

```
>>> l=((1,2,3),(4,5,6),(7,8,9))
>>> l[1]
```

12.

```
>>> l=((1,2,3),(4,5,6),(7,8,9))
>>> l[0][2]
```

## B. Listes ou tables :

### Ce qu'il faut savoir :

- **Une liste ou une table** est une suite ordonnée d'éléments avec des indices pour les repérer. Les éléments d'une liste sont séparés par des virgules et entourés de crochets.

**Exemples :** - table1=[12,23,10,46,9] ou table2= ["a","b","c","d"] ou table3=[True,5,"e"]  
ou table4=[(12, ( 23, 10, 46, 9 ))].  
- table5=[] est **la liste vide**.

- 3 façons de **créer une liste** : - Explicitement (comme les exemples précédents).  
- Par compréhension avec les méthodes **list** ou **range** et avec **for** :  
table6=list(range(2;20;3)) vaut [2,5,8,11,14,17].  
table7=[3\*i for i in range(100)] vaut [0,3,6,9,.....,297].

- Tout ce qui relève de **la concaténation** (avec « + » ou « \* »), de **l'appartenance** (avec « in »), de **la taille** (avec « len ») ou **des indices** est similaire avec les tuples.

- **Remarque :** Les éléments d'une liste existante sont modifiables par une affectation du type liste[i]=...  
Par exemple si liste=[1,4,8], après l'affectation liste[1]=0, liste vaut [1,0,8].

**Exercice 2 :** Pour chaque cas, détermine ce que va afficher Python (après avoir répondu, corrige toi en tapant les instructions sur la console de Python).

1.
 

```
>>> L=[15,17,25,23]
>>> L[2]=25
>>> L
```
2.
 

```
>>> L=[[1,2,3],[4,5,6],[7,8,9]]
>>> L[1]
```
3.
 

```
>>> L=[[1,2,3],[4,5,6],[7,8,9]]
>>> L[1][0]
```
4.
 

```
>>> L=[[i,i+1] for i in range(3)]
>>> L
```
5.
 

```
>>> L=list(range(1,200,50))
>>> L
```
6.
 

```
>>> L1=[5*i for i in range(2)]
>>> L2=[2*i+1 for i in range(3)]
>>> 2*L1+L2
```
7.
 

```
>>> from random import randint
>>> L=[randint(0,10) for i in range(6)]
>>> L
```

### Exercice 3 :

1. Sur la console Python crée la liste des notes au dernier contrôle de nsi.  
L=[14, 19, 15, 14, 9, 9, 9, 19, 14, 9, 17, 16, 8, 16, 8, 13].
2. a. Sur la console, tape « **L.reverse()** » puis « L » pour afficher la liste L.  
b. Déduis-en l'utilité de la méthode « reverse() ».
3. a. Sur la console, tape « **L.sort()** » puis « L » pour afficher la liste L.  
b. Déduis-en l'utilité de la méthode « sort() ».
4. a. Sur la console, tape « **L.count(9)** » puis « **L.count(19)** puis « **L.count(5)** ».  
b. Déduis-en l'utilité de la méthode « count(i) ».
5. a. Sur la console, tape « **del(L[2])** » puis « L » pour afficher la liste L.  
b. Sur la console, tape « **del(L[6])** » puis « L » pour afficher la liste L.  
c. Déduis-en l'utilité de la méthode « del(i) ».
6. a. Sur la console, tape « **L.append(20)** » puis « L » pour afficher la liste L.  
b. Déduis-en l'utilité de la méthode « append(i) ».
7. a. Sur la console, tape « **L.insert(5,10)** » puis « L » pour afficher la liste L.  
b. Déduis-en l'utilité de la méthode « insert(i,j) ».

**Ce qu'il faut savoir :** Les méthodes reverse() / sort() / count(i) / del(L[i]) / append(i) / insert(i,j) sur Python.

	utilité	exemples
reverse		
sort		
count		

<b>del</b>		
<b>append</b>		
<b>insert</b>		

#### Exercice 4 :

1. Ecris une fonction **multiples1** qui prend en paramètre un entier naturel non nul **n** et renvoie la liste des dix premiers multiples non nuls de **n**. Teste la dans la console.
2. Ecris une fonction **multiples2** qui prend en paramètre un entier naturel non nul **n** et renvoie la liste des multiples de **n** inférieurs strictement à 1000. Teste la dans la console.
3. Ecris une fonction **diviseurs** qui prend en paramètre un entier naturel non nul **n** et renvoie la liste des diviseurs de **n**. Teste la dans la console.

#### Exercice 5 : Voici une fonction mystere :

```
def mystere(liste1, liste2) :
    liste = []
    i, j = 0, 0
    while i < len(liste1) and j < len(liste2):
        if liste1[i] < liste2[j] :
            liste.append(liste1[i])
            i = i + 1
        else :
            liste.append(liste2[j])
            j = j + 1
    return liste
```

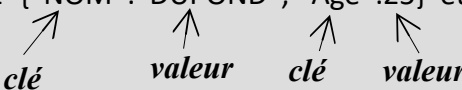
Si on tape l'instruction **mystere([2, 5, 6, 8], [1, 4, 7, 8, 9])** dans la console, quel est le résultat renvoyé ?  
 Trouve le résultat avant de tester sur machine !

**Exercice 6 :** Ecris une fonction qui prend en argument un tuple composé d'entiers et renvoie deux listes : la première liste contient les nombres pairs et la seconde les nombres impairs .  
 Teste cette fonction dans la console.

## Ce qu'il faut savoir :

- **Un dictionnaire** en Python est une suite d'éléments dont les indices sont remplacés par des objets du type str, int, float, tuple, dict : on les appelle **des clés**.  
Chaque clé est associée à une valeur et les clés **ne sont pas ordonnées**.

**Exemples :**


- d2={"NOM":"DUPOND", "Age":25} et d1={"Age":25 , "NOM":"DUPOND"}  
  
clé                  valeur        clé        valeur                  représentent le même dictionnaire.

- d2={("NOM","Prénom":{"DUPOND", "Jean"}, "Age":25}.

### Ce qu'il faut savoir :

- **Un dictionnaire** en Python est une suite d'éléments dont les indices sont remplacés par des objets du type str, int, float, tuple, dict : on les appelle **des clés**.

Chaque clé est associée à une valeur et les clés *ne sont pas ordonnées*.

**Exemples :** - d2={"NOM": "DUPOND", "Age": 25} et d1={"Age": 25, "NOM": "DUPOND"}  
  
 représentent le même dictionnaire.

```
- d2={"NOM","Prénom":("DUPOND", "Jean"), "Age":25}.
```

### Exercice 7 :

1. Sur la console Python crée le dictionnaire ci-dessous.  
D={"NOM" :....., "Prénom" :....., "Age" :....., "Moy\_nsi\_T1\_T2":(.....,.....)}.
2. **a.** Sur la console, tape « **D.keys()** ».      **b.** Déduis-en l'utilité de la méthode « keys() ».
3. **a.** Sur la console, tape « **D.values()** ».      **b.** Déduis-en l'utilité de la méthode « values() ».
4. **a.** Sur la console, tape « **D.items()** ».      **b.** Déduis-en l'utilité de la méthode « items() ».
5. **a.** Sur la console, tape « **D.get("NOM")** » puis « **D.get("Age")** ».      **b.** Déduis-en l'utilité de la méthode « get(i) ».
6. **a.** Sur la console, tape « **list(D.items())** ».      **b.** Déduis-en l'utilité de la méthode « list(i) ».
7. **a.** Sur la console, tape « **del(D[Age])** » puis « D » pour afficher le dictionnaire D.      **b.** Déduis-en l'utilité de la méthode « del(i) ».

**Ce qu'il faut savoir :** Les méthodes reverse() / sort() / count(i) / del(L[i]) / append(i) / insert(i,j) sur Python.

	utilité	exemples
keys		
values		
items		
get		

list		
del		

**Exercice 8 :** Pour chaque cas, détermine ce que va afficher Python (après avoir répondu, corrige toi en tapant les instructions sur la console de Python).

1.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> 'A' in D.keys()
```
2.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> 3 in D.values()
```
3.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> ('B',1) in D.items()
```
4.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> len(D)
```
5.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> for i in D.keys():
...     print(i)
```
6.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> for i in D.values():
...     print(i)
```
7.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> for i in D.items():
...     print(i)
```
8.
 

```
>>> for (i,j) in D.items():
...     print(i,":",j)
```
9.
 

```
>>> D={'A':0, 'B':1, 'C':2}
>>> D['D']=3
>>> D
```

**Exercice 9 :** Soit le dictionnaire **vols** représentant les vols dans un aéroport.

```
vols = {'Lisbonne': {'compagnie': 'EASYJET', 'heure': '21:10', 'num': 'EJU7674'},
        'Londres': {'compagnie': 'BRITISH AIRWAYS', 'heure': '21:55', 'num': 'BA357'},
        'Vienne': {'compagnie': 'AUSTRIAN AIRLINES', 'heure': '21:25', 'num': '05430'}}
```

1. Ecris un script Python pour :
  - a. Afficher les informations du vol vers Lisbonne.
  - b. Afficher les informations du vol vers Vienne.
  - c. Afficher la compagnie chargée du vol vers Londres.
2. Ecris un script pour ajouter un vol vers Paris avec les informations suivantes :
 

heure : 22 : 00  
 num : KJ023  
 compagnie : AIR France