Python : listes et tuples (n-uplets)

Les tuples

Les tuples sont des **listes immuables** c'est à dire que l'on **ne peut pas modifier leur contenu après leur création**. On ne peut plus y ajouter d'objet ou en retirer. Les tuples se définissent en utilisant des **parenthèses** comme délimiteur.

Création d'un tuple

```
tuple_vide = ()
tuple_avec_plusieurs_valeurs = (1, 2, 5)
```

Usage des tuples

- affectation multiple :

```
a, b = 3, 4
(a, b) = (3, 4) # équivalent à l'écriture précédente
```

- permettre à une fonction de renvoyer plusieurs valeurs :

```
def division(entier, divise_par):
    #instructions à mettre pour calculer le quotient et le reste
    return quotient, reste
```

Les opérations de calcul

Opérations	Symboles	Exemples
addition	+	1 + 3 donne 4
soustraction	-	10 - 5 donne 5
multiplication	*	3 * 5 donne 15
exponentiation (puissance)	**	2 ** 4 donne 16
division	/	9 / 2 donne 4.5
reste de division entière	%	5 % 2 donne 1
quotient de division entière	//	5 // 2 donne 2

Exercice sur les tuples

Ecris un programme :

- qui **demande** à l'utilisateur de saisir deux nombres, le dividende et le diviseur,

- qui **appelle** la fonction division() avec en **paramètre** ces deux nombres,
- qui **renvoie** le quotient et le reste de la division euclidienne de ces deux nombres.

Les listes

Les listes sont des objets qui peuvent contenir d'autres objets de n'importe quel type tel que des entiers, des réels, des chaînes de caractères.

Les **listes** du langage Python correspondent aux **tableaux** dans d'autres langages de programmation.

Création d'une liste

Il y plusieurs manières de créer une liste :

```
ma_liste1 = list()                # création d'une liste vide à partir du type list
ma_liste2 = []                    # création d'une liste vide
ma_liste3 = [0]*5                 # création d'une liste contenant 5 cases avec la valeur 0
ma_liste4 = [0 for i n range(5)]  # création d'une liste contenant 5 cases avec la valeur 0
ma_liste5 = [1,2,3,"ICN",[ ]]     # création d'une liste contenant 5 cases avec des données de type différent
```

Voici le résultat de ces différentes créations de listes :

Nom	Type	Taille	Valeur
ma_liste1	list	0	[ ]
ma_liste2	list	0	[ ]
ma_liste3	list	5	[0,0,0,0,0]
ma_liste4	list	5	[0,0,0,0,0]
ma_liste5	list	5	[1,2,3,'ICN', [ ]]

Accès à une case d'une liste

- Accès en lecture :

```
In []: ma_liste5[3]
Out[]: 'ICN'
```

- mise à jour d'une case

```
In []: ma_liste5[3] = 'Informatique'
```

Le contenu de la liste a changé :

Nom	Type	Taille	Valeur
ma_liste5	list	5	[1,2,3,'Informatique', [ ]]

Opérations sur les listes

Description	Instruction python	Contenu de ma_liste5	Remarque
Ajouter un élément à la fin de la liste	ma_liste5.append(56)	[1,2,3,'Informatique', [ ], 56]	
Insérer un élément dans la liste	ma_liste5.insert(1, “S”)	[1,'S',2,3,'Informatique', [ ], 56]	La lettre **S** a été insérée à l'indice 1
Supprimer un élément dans la liste	ma_liste5.remove(“S”)	[1,2,3,'Informatique', [ ], 56]	Il faut préciser l'élément à supprimer et non pas son indice
Rechercher un élément dans la liste	if element in ma_liste5:		
Fournir l'emplacement d'un élément dans la liste	ma liste5.index(element)	renvoie 3 à ma liste5.index(2)	

- Parcourir une liste

```
for element in ma_liste5:
   print(element)
```

La boucle for permet de **parcourir** la liste et la variable **element** va prendre successivement les valeurs de chaque élément de la liste.

- Parcourir une liste avec la méthode enumerate

```
for element in enumerate(ma_liste5):
   print(element)
Cela renvoie
(0, 1)
(1, 2)
(2, 3)
(3, 'Informatique')
(4, [])
(5, 56)
```

La boucle for permet de **parcourir** la liste et la variable **element** va prendre successivement un couple de valeurs constitué de l'indice et de la valeur de chaque élément de la liste. Il s'agit d'un **tuple** (n-uplets)

Exercice1

Voici le programme qui permet de lancer 12 fois un dé et de compter le nombre de fois que l'on a tiré 1,2,3...

```
import random
k1=k2=k3=k4=k5=k6=0
for i in range(1,13):
        de = random.randint(1,6)
        print (de)
        if de==1:
                k1=k1+1
        elif de==2:
                k2=k2+1
        elif de==3:
                k3=k3+1
        elif de==4:
                k4=k4+1
        elif de==5:
                k5=k5+1
        elif de==6:
                k6=k6+1
print ("le nombre de ",1,"est ",k1)
print ("le nombre de ",2,"est ",k2)
print ("le nombre de ",3,"est ",k3)
print ("le nombre de ",4,"est ",k4)
print ("le nombre de ",5,"est ",k5)
print ("le nombre de ",6,"est ",k6)
```

Essayez de réaliser le même programme en utilisant des tableaux.

Exercice 2

Écrire un programme qui trouve la plus grande note dans une liste de 10 notes.

Il faudra d'abord saisir les 10 notes. Pour cela, utilisez l'instruction permettant de saisir une valeur numérique que vous ajouterez dans le tableau.

Exercice 3

Ecris un programme répertoire permettant de donner le numéro de téléphone d'une personne à partir de son prénom. Pour cela, tu vas associer des numéros de téléphone à des prénoms en utilisant deux listes :

- la première liste **prenom** contiendra les prénoms de telle sorte que le prénom d'une personne soit accessible par **prenom[i]**,
- la deuxième liste **tel** contiendra le numéro de téléphone de sorte que le numéro **tel[i]** soit associé à la personne **prenom[i]**.

Ton programme doit contenir la liste de 10 prénoms et permettre de donner le numéro de téléphone d'un prénom que l'utilisateur va saisir. On suppose que les données ne seront pas saisies par l'utilisateur, mais directement entrées dans le programme.

Exercice 4

Ecrire un programme qui permet la saisie de 4 éléments:
le numéro de jour dans la semaine
le numéro de jour dans le mois
le numéro de mois
l'année
Après avoir contrôlé la valeur de ces 4 données, vous ferez afficher le nom du jour dans la semaine, le numéro du jour, le mois, et l'année.
Pour l'année, si sa valeur est inférieure à 17, c'est une année 2000, sinon, c'est une année 1900.

Exemple de saisie: 3, 10, 5, 16
Affichage correspondant: Mercredi 10 mai 2016.

Les listes bidirectionnelles

Pour représenter une table à double entrée, on peut utiliser les listes de la manière suivante comme pour gérer ce tableau de 4 lignes de 3 colonnes.

Trimestre	Français	Maths
1	12	15
2	14	16
3	18	12

Création de la liste **notes** avec les différentes cases contenant la valeur 0:

```
notes = [ [00 for j in range(4)]  for i in range(3)]
```

La liste notes se présente ainsi :

Nom	**Type**	**Taille**	**Valeur**
notes	list	3	[[0,0,0,0], [0,0,0,0], [0,0,0,0]]

La case de la colonne **i** et la ligne **j** de la liste notes est désignée par l'expression **notes[i][j]**. Pour mettre la note 15 en Maths au premier trimestre on écrit l'instruction suivante :

```
notes[2][1] = 15
```

La liste **notes** est modifiée :

Nom	Type	Taille	Valeur
notes	list	3	[[0,0,0,0], [0,10,0,0], [0,15,0,0]]

Retour au cours : Les instructions du langage Python

- Cours : Les instructions du langage Python