Les Tableaux et Dictionnaires en Python

I- Création d'un tableau sans utiliser de bibliothèque :

On peut créer un tableau comme indiqué ci-dessous

```
print ("Tableau basic avec python 3 -sans numpy- ")


t= [[1,2,3,4,5] ,
    [6,7,8,9,0] ,
    [0,0,0,0,0] ]



for y in range (3):
    for x in range(5):
        print (t[y][x] , end =" ")
    print ()
```

L'astuce c'est qu'en fait les tableaux sont des listes de listes. Si on veut créer un tableau de dimension N1*N2 quelconque, on sera obligé d'écrire une fonction spécifique. L'utilisation de la bibliothèque Numpy (qu'on verra plus loin) permet de créer et manipuler plus simplement des tableaux et on ne s'en privera pas !

II- Création d'un dictionnaire

Un dictionnaire en python est une liste de couples (clef : valeur)

```
print ("Dictionnaire avec python ")
data = {}   # initialiser dictionnaire

data = {"Jean Pierre" : 25 , "Bertrand" : 49 , "Jérome" : 50 , "David" : 12}
```

CLEF VALEUR

Note : clef et valeur peuvent être du texte ou des nombres

Manipulation du dictionnaire :

```
# Ajouter des valeurs à un dictionnaire

data["Isabelle"] = 35


# Affichage de la valeur correspondante à la clef
print ("Age de Bertrand ", end =" : ")
print (data.get("Bertrand"))

# Affichage de toutes les clefs :

for clef in data.keys():
    print (clef)

# affichage des valeurs :

for valeur in data.values():   # l'ordre d'écriture en sortie varie
    print (valeur)

# affichage des couples  clef,valeur

for clef,valeur in data.items():
    print (clef,valeur)

# Recherche d'une clef dans un dictionnaire

if "Bertrand" in data:
    print ("La clef Bertrand est bien présente")
```

On testera le code pour s'approprier la notion de dictionnaire --> on rajoutera des couples de clefs/valeur

Recherches suivant des critères: Il n'y a pas de fonction spécifique, il faut donc les programmer. Par exemple, recherche en fonction critère d'âge :

```
# Je veux rechercher toutes les personnes de plus de  30 ans:
print ()
print ("Voici les personnes de plus de 30 ans: ")
for clef,valeur in data.items():
    if valeur>30:
        print (clef)

print ()

# Supprimer une clef (et valeur associée) à un dictionnaire
del data["Isabelle"]
if "Isabelle" in data:
    print ("La clef Isabelle  est bien présente")
else:
    print ("La clef Isabelle  n'existe plus")
```

Que se passe t-il si on essaie de supprimer une clef qui n'existe pas ?

Que faut-il donc faire au préalable ?

III- Les tableaux avec la bibliothèque Numpy

L'utilisation des tableaux étant particulièrement courante et pratique dans de nombreuses applications et en particuliers les jeux, on va voir comment en créer de façon très simple avec la bibliothèque Numpy :

```
# On veut afficher un tableau de 8 lignes (y) et 10 colonnes (x)

from numpy import *     # Chargement de la bibilothèque numpy


# Procédure d'affichage du tableau

def affiche_tableau(x1,y1):
    for y in range(y1):
        for x in range(x1):
            print (t[y,x], end=" ")
        print ()


nb_ligne = 8
nb_colonne = 10

# création du tableau

t=zeros([nb_ligne,nb_colonne],int) # t[y,x] y=8 lignes et x=10 colonnes

# Ci dessous possibilité d'initialiser le tableau

t[0,] = [1,1,1,1,1,1,1,1,1,1]
t[1,] = [1,2,0,0,0,0,0,0,0,1]
t[2,] = [1,0,1,1,1,1,1,0,0,1]
t[3,] = [1,0,0,0,0,0,1,0,0,1]
t[4,] = [1,0,1,1,1,0,1,1,1,1]
t[5,] = [1,0,0,0,1,0,1,0,0,1]
t[6,] = [1,0,0,0,1,0,0,0,3,1]
t[7,] = [1,1,1,1,1,1,1,1,1,1]

affiche_tableau(nb_colonne,nb_ligne) # appel de la procédure d'affichage
```

```
1 1 1 1 1 1 1 1 1 1
1 2 0 0 0 0 0 0 0 1
1 0 1 1 1 1 1 0 0 1
1 0 0 0 0 0 1 0 0 1
1 0 1 1 1 0 1 1 1 1
1 0 0 0 1 0 1 0 0 1
1 0 0 0 1 0 0 0 3 1
1 1 1 1 1 1 1 1 1 1
```

Création du tableau t en mémoire

On peut adapter l'affichage du tableau suivant les besoins en modifiant légèrement la procédure d'affichage :

```
def affiche_tableau2(x1,y1):
    for y in range(y1):
        for x in range(x1):
            if t[y,x] == 0:
                print (".", end=" ")
            else:
                print (t[y,x], end=" ")

        print ()
```

```
1 1 1 1 1 1 1 1 1 1
1 2 . . . . . . . 1
1 . 1 1 1 1 1 . . 1
1 . . . . . 1 . . 1
1 . 1 1 1 . 1 1 1 1
1 . . . 1 . 1 . . 1
1 . . . 1 . . . 3 1
1 1 1 1 1 1 1 1 1 1
```