

TP10 : tableaux magiques

Partie A : sur papier

1) On donne l'algorithme d'une fonction mystere(tab) qui prend en paramètre un tableau.

```
fonction mystere(tab) :  
    variables locales :  
        test de type booléen  
        resu de type tableau  
        i de type entier  
début fonction  
    test ← VRAI  
    POUR i allant de 0 à longueur(tab)-1 :  
        SI tab[i] ≠ tab[0] :  
            test ← FAUX  
    RETOURNER test  
fin fonction
```

1.a) Quelle valeur est retournée par la fonction si on exécute :
mystere([4,1,8])

1.b) Quelle valeur est retournée par la fonction si on exécute :
mystere([4,4,4])

1.c) Quel est le rôle de la fonction mystère ?

2) On donne le code python suivant

```
carre = [[1,2,3],  
         [4,5,6],  
         [7,8,9]]  
  
def total_par_ligne(tab):  
    n = len(tab)  
    m = len(tab[0])  
    resu = [0 for i in range(n)]  
    for i in range(n):  
        for j in range(m):  
            resu[i] = resu[i] + tab[i][j]  
    return resu
```

On exécute la fonction : total_par_ligne(carre)

2.a) Compléter le tableau de suivi des variables locales

tab	n	m	resu	i	j
[[1,2,3], [4,5,6], [7,8,9]]					

2.b) Quelle sera la valeur retournée par la fonction ?

Partie B : sur ordinateur

On donne le code python suivant (fichier TP10.py)

```
carre = [[1,2,3],
         [4,5,6],
         [7,8,9]]

rectangle = [[1,4,6,7],
             [8,5,3,2]]

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

grandcarre = [[12,6,15,1],
               [13,3,10,8],
               [2,16,5,11],
               [7,9,4,14]]

def mystere(tab):
    test = True
    for i in range(len(tab)):
        if tab[i] != tab[0]:
            test = False
    return test

def total_par_ligne(tab):
    n = len(tab)
    m = len(tab[0])
    resu = [0 for i in range(n)]
    for i in range(n):
        for j in range(m):
            resu[i] = resu[i] + tab[i][j]
    return resu
```

1) Tester vos réponses données dans la partie A.

2) Ecrire une fonction `total_par_colonne(tab)` : qui prend en argument un tableau rectangulaire à deux dimensions, et qui renvoie un tableau à une dimension contenant le total de chaque colonne de `tab`.

Par exemple :

`total_par_colonne(carre)`

doit retourner

`[12, 15, 18]`

car le tableau est :

```
carre = [[1,2,3],
         [4,5,6],
         [7,8,9]]
```

et

12 est le total de la première colonne $1+4+7$

15 est le total de la deuxième colonne $2+5+8$

18 est le total de la troisième colonne $3+6+9$

3) Un rectangle magique est un tableau dont la somme des valeurs en lignes est toujours la même, et dont la somme des valeurs de colonnes est toujours la même : vérifier, en utilisant les fonctions programmées, que le tableau `rectangle` est magique.

4) Programmer une fonction `magique(tab)` qui renvoie vrai si le tableau `tab` passé en paramètre est « magique », c'est à dire que la somme de toutes les lignes est identique, et la somme de toutes les colonnes également

Pour aller plus loin :

dans le cas d'un tableau carré, on peut tester la somme de chaque diagonale.

5.a) Ecrire deux fonctions `diago1(tab)` et `diago2(tab)` qui calculent, lorsque `tab` est un tableau carré, la somme des termes sur chaque diagonale.

5.b) Ecrire une fonction `carre_magique(tab)` qui teste, en plus des lignes et des colonnes, les diagonales d'un carré pour savoir s'il est magique.