ACADÉMIE
DE REIMS
Liberté
Égalité
Fraternité

NUMERIQUE
et
SCIENCES
INFORMATIQUES

Épreuve de l'enseignement de spécialité

Sujet d'entraînement

Partie Pratique

Classe Terminale de la voie générale

Le candidat doit traiter les 2 exercices

Ce sujet comporte 2 pages

Exercice 1 (4 points)

Programmer une fonction `min_max_somme` qui prend en paramètre une liste non vide de nombres entiers `tab` et qui renvoie un dictionnaire ayant pour clés `"Min"`,`"Max"`,`"Somme"` et pour valeur associée respectivement la valeur du minimum, du maximum et de la somme des éléments de la liste.

Exemple :

```
>>> assert min_max_somme([4,10,0,8]) == {'Min': 0, 'Max': 10, 'Somme': 22}
```

Exercice 2 (4 points)

On souhaite programmer une fonction récursive `fusion_rec` qui prend en paramètres deux tableaux non vides `tab1` et `tab2` (type `list`) d'entiers, chacun trié dans l'ordre croissant, et qui renvoie un tableau trié dans l'ordre croissant et contenant l'ensemble des valeurs de `tab1` et `tab2`.

Dans ce code, vous pouvez utiliser `slicing`. Voici quelques exemples pour comprendre le `slicing` en python :

```
>>> tab=[2,6,5,7,8]
>>> tab[2:4]
[5, 7]
>>> tab[:4]
[2, 6, 5, 7]
>>> tab[2:]
[5, 7, 8]
```

Voici le code à compléter :

```
def fusion_rec(gauche,droite):
    if gauche==[] :
        return ...
    elif droite==[] :
        return ...
    else :
        if ... <= ...:
            return [...]+fusion_rec(gauche[1:],...)
        else :
            return [droite[0]]+...(...,droite[1:])
```

Un jeu de tests à votre disposition :

```
>>> assert fusion_rec([1, 2, 8, 9 ], [2, 12]) == [1, 2, 2, 8, 9, 12]
>>> assert fusion_rec([-2, -1, 0], [-7, -2, -1]) == [-7, -2, -2, -1, -1, 0]
>>> assert fusion_rec([8], [1, 9]) == [1, 8, 9]
```