ISM MPSI/PCSI, Informatique
Année 2024/2025

Corrigé du TP Informatique 29

Exercice 1

1. On saisit :

```
def fusion(T,g,m,d):
    """fusion(T:list,g:int,m:int,d:int)->None
    Réalise la fusion de T[g:m] et T[m:d] en modifiant T"""
    tab=T[g:m]+T[d-1:m-1:-1]
    i=0
    j=-1
    for k in range(g,d):
        if tab[i]<tab[j]:
            T[k]=tab[i]
            i+=1
        else:
            T[k]=tab[j]
            j-=1
```

2. On saisit :

```
def tri_fusion_rec(T,g,d):
    """tri_fusion_rec(T:list,g:int,d:int)->None
    Réalise le tri fusion de T[g:d] en modifiant T"""
    if d-g>1:
        m=(g+d)//2
        tri_fusion_rec(T,g,m)
        tri_fusion_rec(T,m,d)
        fusion(T,g,m,d)
```

3. On saisit :

```
def tri_fusion(T):
    """tri_fusion(T:list)->None
    Réalise le tri fusion de T en modifiant T"""
    tri_fusion_rec(T,0,len(T))
```

4. Il ne s'agit pas d'un tri en place car la variable `tab` créée par la fonction `fusion` est de même taille que l'argument d'entrée après les derniers appels récursifs.

Exercice 2

1. On saisit :

```
def tri_comptage(L):
    """tri_comptage(L:list)->list
    Réalise le tri par comptage d'une liste d'entiers"""
    maxi=L[0]
    # recherche du maximum
    for x in L[1:]:
        if x>maxi:
            maxi=x
    # tab : liste des nbs d'occurrences des entiers de 0 à maxi
    tab=[0]*(maxi+1)
    for x in L:
        tab[x]+=1
    res=[]
    # construction du résultat
    for k  in range(maxi+1):
        # répétition d'un entier, autant de fois que son nb d'occurrences
        res+=[k]*tab[k]
    return res
```

2. Avec la liste `[0,2**2025]`, le programme commence par la construction de la liste `[0]*2**2025` ce qui est particulièrement inefficace. Le tri par comptage est pertinent pour une liste de nombres qui ne sont pas trop étalés.