Tri par comptage

(pour des entiers positifs)

```
def TriComptage(T) :
    M = Max(T)          : recherche de la valeur max de T (cf algo du min)
    Cpt = [0… 0]              :  tableau de taille M+1 pour compter
    Pour i= 0 … |T|-1 :
            Cpt[T[i]]++
    res = [0… 0]              :  tableau de taille |T|
   j=0
   Pour i=0..M :
       “ajouter Cpt[T[i]] valeur T[i] depuis j”
       j = j+Cpt[T[i]]

   Retourner res
```

NB: la recherche du Max et l’ajout de valeurs dans res a un coût… il faut écrire ces algo…

Tri par comptage

(pour des entiers positifs)

Version améliorée

```
def TriComptage(T) :
    M = Max(T)
    Cpt = [0… 0]            :  tableau de taille M+1
    Pour i= 0 … |T|-1 :
            Cpt[T[i]]++
    Pour k=1…M :
            Cpt[k] = Cpt[k] + Cpt[k-1]
    res = [0… 0]            :  tableau de taille |T|
    Pour i =    |T|-1…0 :
            res[Cpt[T[i]]-1] = T[i]
            Cpt[T[i]] -= 1
    Retourner res
```

ici Cpt[k] contient le nb de valeurs ≤ k

Tri drapeau

```
# z1 est la première case qui n'est pas dans la zone des 1
# z2 idem pour les 2
# z3 est la dernière case avant la zone des 3
def TriDrapeau(T) :
    z1=0,     z2=0,     z3=len|T|-1
    while z2 <= z3 :
        if T[z2] == 2 :  z2 += 1
        else if T[z2] == 3 :
                T[z3],T[z2] = T[z2],T[z3]
                z3-=1
        else :
            T[z1],T[z2] = T[z2], T[z1]
            z1+=1
            z2+=1
return T
```