E.D. NFP 136 n°6

Thème : Tris

Exercice 1 Le tri fusion

Appliquer en détail l'algorithme du tri fusion à la suite de nombres suivante, pour les trier par ordre *décroissant* :

16 - 11 - 9 - 10 - 5 - 6 - 8 - 1 - 2 - 4

Exercice 2 Le tri rapide

Soit le tableau d'entiers suivant :

150	200	100	300	80	30	50	40
0	1	2	3	4	5	6	7

Appliquer l'algorithme de tri rapide pour trier ces entiers par ordre croissant.

Exercice 3 Le tri par dénombrement

Si les valeurs à trier sont « petites » (en comparaison du nombre de valeurs), on peut utiliser un autre type de tri que les tris par comparaison, appelé tri par dénombrement.

L'idée de ce tri n'est pas de comparer les valeurs entre elles, mais de compter le nombre de fois où chaque valeur (comprise entre 0 et Vmax) apparaît. En voici un code Java :

```
int[] triParDenombrement  (int[] tabATrier) {
int i, val, n=tabATrier.length, Vmax=tabATrier[0], compteur=0;
for(i=1;i<=n-1;i++) {if(tabATrier[i] > Vmax) Vmax=tabATrier[i];}
//tabATrier=tableau de n valeurs à trier, toutes comprises entre 0 et Vmax
int[] card=new int[Vmax+1]; //card[v]=nombre d'éléments de tabATrier égales à v
int[] tabTrie=new int[n]; //tabTrie=tableau trié contenant les éléments de tabATrier

for(val=0 ; val<=Vmax ; val++)  {card[val]=0;}
for(i=0 ; i<=n-1 ; i++) {card[tabATrier[i]]++;}
for(val=0 ; val<=Vmax ; val++) {
        for(i=0 ; i<=card[val]-1 ; i++) {tabTrie[compteur+i]=val;}
        compteur+=card[val];
}
return tabTrie;
}
```

Question 1

Montrer que la complexité au pire cas de cet algorithme est en O(n+Vmax) (et donc en O(n) si Vmax=O(n)), puis donner sa complexité en mémoire.

Question 2

Appliquer cet algorithme sur la liste de nombres suivante : 2-5-3-0-2-3-0-3