Université Larbi Ben M'hidi

Faculté : SESNV

Département : Mathématiques et Informatique

Module : ASD 3

Semestre : 3 (Informatique) Janvier 2023

Corrigé du contrôle (2022/2023)

Exercice 1 (05 pts):

Considérer deux algorithmes A1 et A2 avec leurs temps d'exécution respectifs $T_1(n) = 100n$ et $T_2(n) = 10n^2$.

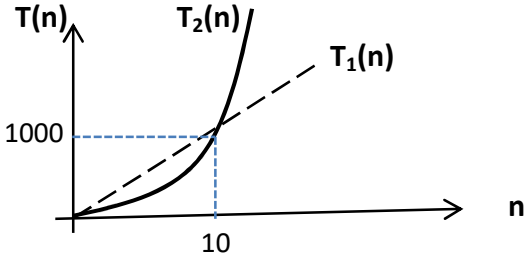
1. Déterminer la complexité asymptotique des deux algorithmes dans la notation Grand-O. Quel algorithme a la meilleure complexité asymptotique ? **1,5 pt**

   $T1(n) = 100n \in O(n)$

   $T2(n) = 10n^2 \in O(n^2)$

   L'algorithme A1 a la meilleure complexité.

2. Ebaucher les graphes des deux fonctions T_i dans un même système de coordonné. **1 pt**



3. Pour quelles longueurs de données n, chacun des algorithmes est le plus efficace ? **1 pt**
   - **0 <n < 10 l'algorithme A2 est plus efficace**
   - **n>10 l'algorithme A1 est plus efficace**

4. Quelle est la complexité asymptotique de l'algorithme suivant ? **1,5 pt**

```
Début
        Si (n <100)alors
                appeler A2 {Ici l'algorithme 2 est exécuté}
        Sinon
                appeler A1 {Ici l'algorithme 1 est exécuté}
        Fin si
fin
```

Les grandes valeurs de n déterminent le temps d'exécution asymptotique → le temps d'exécution de l'algorithme A1.

La Complexité de cet algorithme est *O(n)*

Exercice 2 (05 pts): Rappelons l'algorithme de tri à bulles qui est une certaine forme de tri par sélection duminimum.

```
void TriBulles (int* t, int nbElements) {
   int i, k ;
   for (i = 0; i < nbElements-1; i++) {
      for (k = nbElements-1; k > i; k--) {
         if (t[k] < t[k-1])
             echanger(&t[k], &t[k-1]) ;  }

}}
```

Soit le tableau de caractère suivant : H G F E D C B A

1. Transformez l'algorithme pour qu'il puisse trier des caractères. **0,5pt**
   Il suffit de changer « void TriBulles (*int* t*, int nbElements) » par
   « void TriBulles (*char* t*, int nbElements) »

2. Donnez les états successifs du tableau à la fin de chaque étape de la boucle '***for'*** interne lorsque i = 0. **1pt**

K=7 :	**HGFEDCAB**	**K=3 :**	**HGAFEDCB**
K=6 :	**HGFEDACB**	**K=2 :**	**HAGFEDCB**
K=5 :	**HGFEADCB**	**K=1 :**	**AHGFEDCB**
K=4 :	**HGFAEDCB**		

3. Même question pour la fin de chaque étape de la boucle ***'for'*** externe (principale). **1,5 pt**

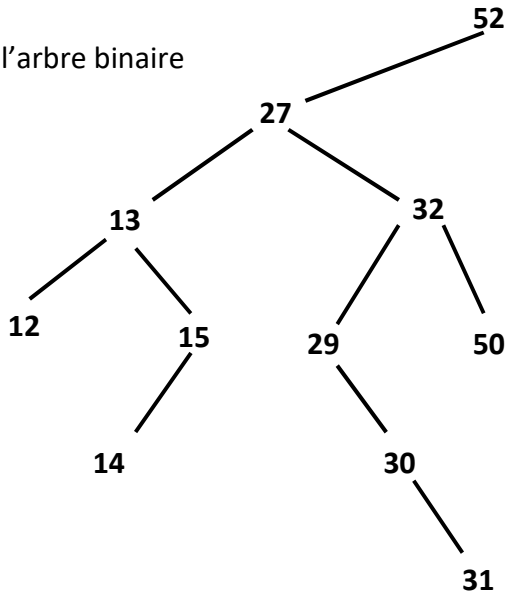
I=1 :	**ABHGFEDC**	**i=4 :**	**ABCDEHGF**
I=2 :	**ABCHGFED**	**i=5 :**	**ABCDEFHG**
I=3 :	**ABCDHGFE**	**i=6 :**	**ABCDEFGH**

4. Donnez l'algorithme de tri à bulles du plus grand élément. Le tri doit se faire toujours par ordre croissant.

```
void TriBulles (int* t, int nb) {
   int i, k ;
   for (i = nb-1; i >0; i--) {                    0,5pt
      for (k = 0; k < i; k++) {                   0,5pt
         if (t[k] > t[k+1])                       0,5pt
            echanger(&t[k], &t[k+1]) ;  }         0,5pt
 }
}
```

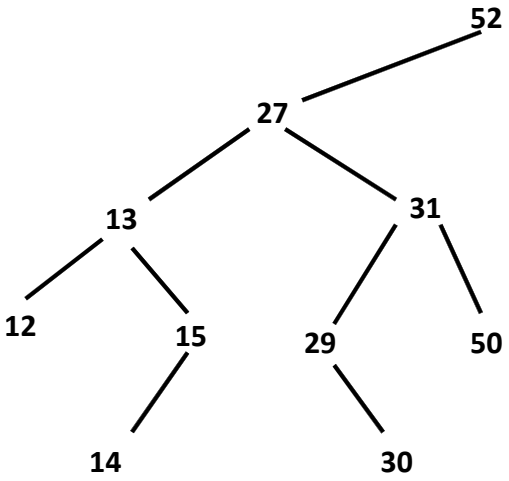
Exercice 3 (06 pts):

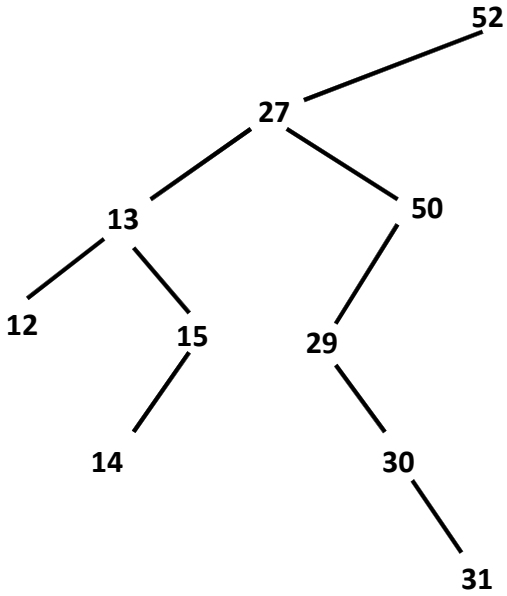
1. Donner le résultat des 3 parcours en profondeur de l'arbre binaire de recherche suivant. **1,5pt**



Prefixe : 52 27 13 12 15 14 32 29 30 31 50
Infixe : 12 13 14 15 27 29 30 31 32 50 52
Postfixe : 12 14 15 13 31 30 29 50 32 27 52

2. Donner les nouveaux arbres obtenus en supprimant l'élément 32. **1pt**

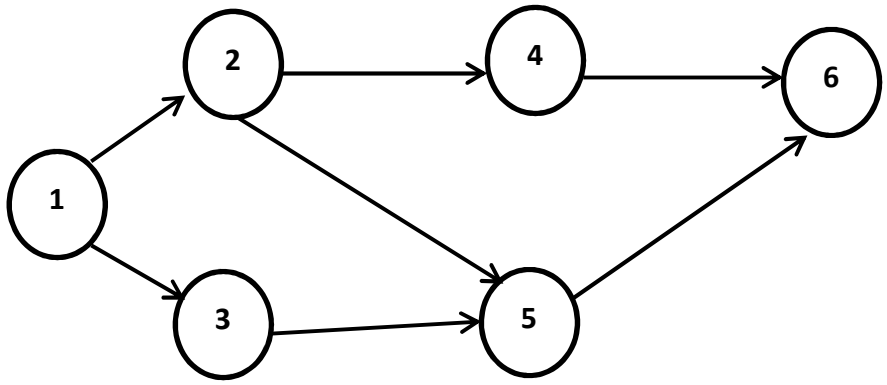




3. Ecrire une fonction récursive **abrEquals(Noeud *R1, Noeud *R2)** comparant 2 arbres binaires de recherche de racines respectives R1 et R2. La fonction retourne true si les 2 arbres sont égaux et false sinon.

```
bool abrEquals(Noeud *R1, Noeud *R2){                          0,5pt
  if (R1==NULL && R2==NULL) return true;                       0.75pt
  else if (R1==NULL || R2==NULL) return false;                 0.75pt
       else if (R1->val==R2->val && abrEquals(R1->filsG,R2->filsG) &&
                abrEquals(R1->filsD, R2->filsD))
                    return true;                               0,5pt+0,5pt+0,5pt
            else return false;
}
```

Exercice 4 (04 pts):

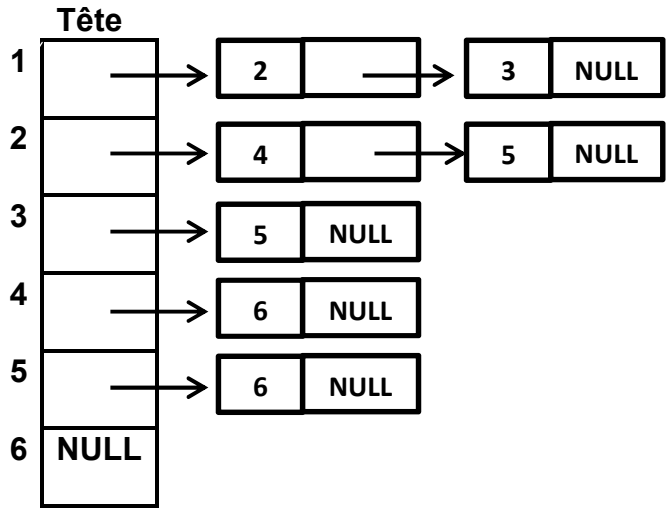


1. Donnez deux représentations de ce graphe. **2pts**

Matrice d'adjacence

	1	2	3	4	5	6
1	0	1	1	0	0	0
2	0	0	0	1	1	0
3	0	0	0	0	1	0
4	0	0	0	0	0	1
5	0	0	0	0	0	1
6	0	0	0	0	0	0

Liste chainée



2. Donnez les résultats du parcours de ce graphe à partir du sommet 1. **2pts**
   Parcours en largeur –BFS- : 1 2 3 4 5 6
   Parcours en profondeur –DFS- : 1 2 4 6 5 3