2LMD **10 Sep 2008**

Examen de Rattrapage ALGO1

8h-9h30 **A2 + S10**

Exercice 1 (8 Pts : 1+1+1+2+3)

Soit une liste linéaire chaînée contenant des nombres entiers et dont la tête est L :

1. Ecrire la fonction **CAR(L)** qui retourne la valeur du premier élément de la liste.
2. Ecrire la fonction **CDR(L)** qui retourne la liste sans le premier élément.
3. Ecrire la fonction **CONS(x, L)** qui retourne une liste dont le premier élément est x et le reste est la liste L.
4. Ecrire la fonction **Triée(L)** qui retourne vrai si la liste L est triée dans l'ordre croissant et faux sinon.
5. Ecrire la fonction **Fusion(L1, L2)** qui prend deux listes triées dans l'ordre croissant L1 et L2 et retourne une liste triée, dans le même ordre, contenant les deux listes et cela en utilisant les fonctions précédentes.

Exercice 2 (12 Pts : 1 + 1.5 + 1.5 + 1 + 2 + 2 + 3)

1. Dessiner l'arbre binaire de recherche obtenu en insérant les éléments de la liste suivante dans leur ordre d'arrivée: 30, 40, 23, 58, 48, 26, 11, 13, 20
2. Donner le résultat des parcours préfixe, infixe et postfixe de l'arbre précédent
3. Donner l'arbre précédent après la suppression des clés suivantes : 48 puis 23.
4. Donner l'arbre précédent (avant la suppression) après l'ajout des clés suivantes : 60 puis 17.
5. Ecrire la fonction **NB_Sup (R :Nœud, x :clé)** qui retourne le nombre de clés supérieures à x dans l'arbre binaire de recherche de racine R.
6. Ecrire la fonction **Val_Sup (R :Nœud, x :clé)** qui retourne la valeur de la plus petite clé supérieure à x dans l'arbre binaire de recherche de racine R.
7. Ecrire la fonction **Fusion(R1,R2 : Nœud)** qui prend deux arbres binaires de recherches de racines R1, R2 et retourne la racine d'un arbre binaire de recherche contenant les deux. Utilisez les deux procédures Suppr(R, clé) et Inser(R, clé) qui permettent respectivement la suppression et l'insertion de clé dans l'arbre de racine R.

Bon courage

A.Djeffal

Corrigé type

Exercice 1

1.

```
Fonction  CAR(L :Pointeur(TMaillon)) :entier ;
Debut
    Si L<>Nil Alors CAR ← Valeur(L) ;
Fin ;
```

2.

```
Fonction  CDR(L :Pointeur(TMaillon)) : Pointeur(TMaillon) ;
Debut
    Si L<>Nil Alors
          CDR ← Suivant(L) ;
           Libérer(L) ;
     Sinon
       CDR← Nil ;
      FSi ;
Fin ;
```

3.

```
Fonction  CONS(x :entier ; L : Pointeur(TMaillon)) : Pointeur(TMaillon);
  Var P : Pointeur(TMaillon) ;
Debut
    Allouer(P) ;
    Aff_Val(P,x) ;
    Aff_Adr(P,L) ;
    CONS← P;
Fin ;
```

4.

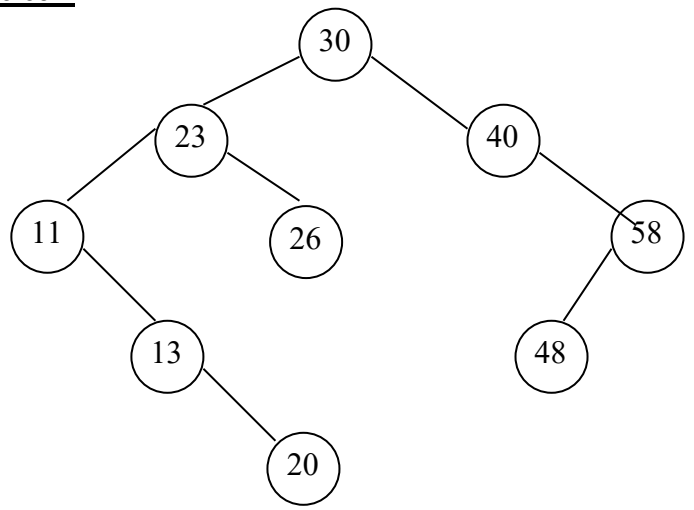
```
Fonction  Triée(L : Pointeur(TMaillon) : Booleen;
Var P : Pointeur(TMaillon) ;
Debut
   P ← L ;
Tantque P<>Nil et Suivant(P)<>Nil et Valeur(P)<Valeur(Suivant(P) Faire
      P ← Suivant(P) ;
FTQ ;
Triée ← ((P=Nil) ou (Suivant(P)=Nil)) :
Fin ;
```

5.

```
Fonction  Fusion(L1,L2 :Pointeur(TMaillon)) : Pointeur(TMaillon) ;
Debut
    Si L1=Nil Alors  Fusion ← L2
    Sinon
      Si L2=Nil Alors  Fusion ← L1
      Sinon
       Si CAR(L1)<CAR(L2) alors
         Fusion ← CONS(CAR(L1), Fusion(CDR(L1),L2))
       Sinon
         Fusion ← CONS(CAR(L2), Fusion(L1,CDR(L2)))
       FSi ;
      Fin ;
    FSi
Fin;
```

Exercice 2

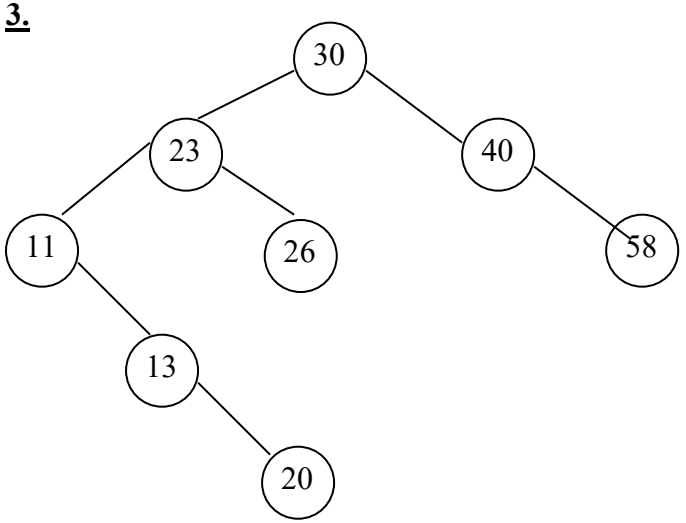
1.

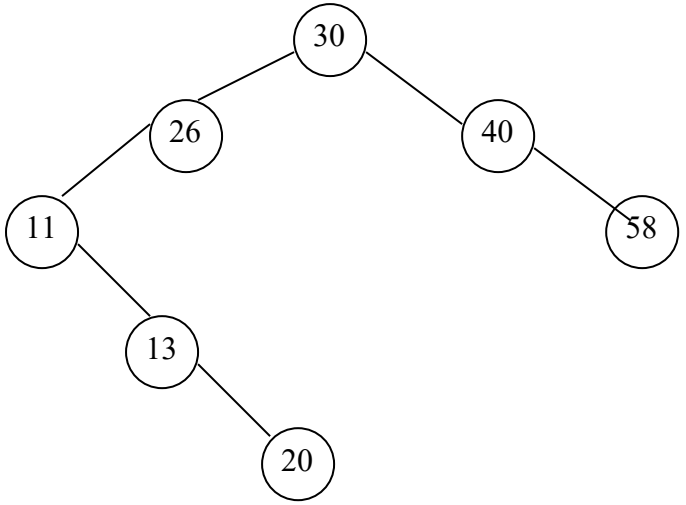


2.

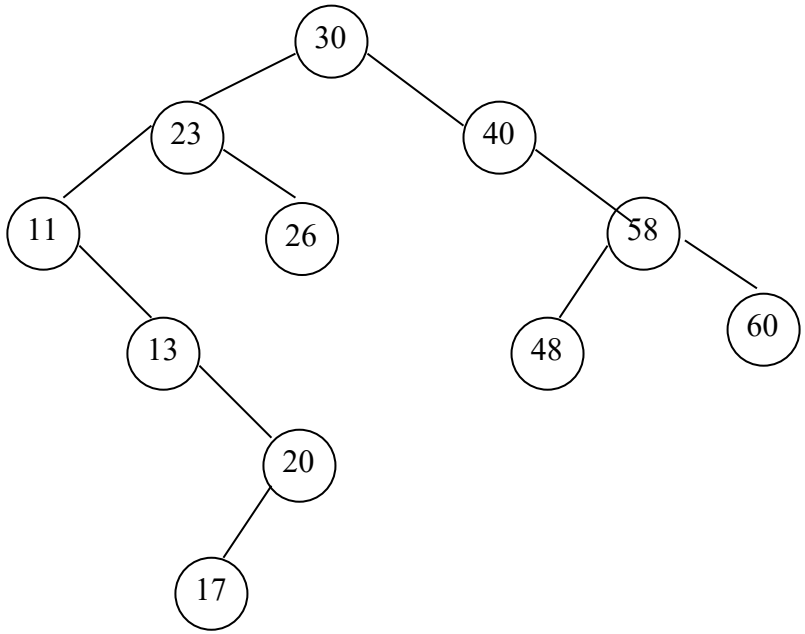
Préfixe : 30 23 11 13 20 26 40 58 48
Infixe : 11 13 20 23 26 30 40 48 58
Postfixe : 11 13 20 26 23 40 48 58 30

3.





4.



5.

```
Fonction   NB_Supp(R :Nœud, x :clé) :entier ;
Debut
   Si R=nil  alors  NB_Supp ← 0
   Sinon
     Si  Clé(R)>x alors
          NB_Supp ← 1 + NB_Supp(FG(R)) + NB_Supp(FD(R))
     Sinon
        NB_Supp ← NB_Supp(FG(R)) + NB_Supp(FD(R))
     FSi
   FSi
Fin;
```

6.

```
Fonction   Val_Supp(R :Nœud, x :clé) :clé ;
Var DG,N : Nœud ;
Debut
   N ← R ;
   TQ N<> Nil et clé(N) <> x Faire
       Si Clé(N)>x  alors  DG ← N ;  N← FG(N)
       Sinon    N← FD(N)
       FSi
   FTQ ;
   Si N=Nil  alors erreur
   Sinon
      N←FD(N);
      Si N=nil alors  Val_Supp ← Clé(DG)
      Sinon
         TQ FG(N) <> nil faire
           N← FG(N) ;
          FTQ
         Val_Supp ← Clé(N) ;
      FSi
    FSi
Fin ;
```

7.

```
Fonction   Fusion(R1,R2 :Nœud) :Noeud ;
Debut
   TQ R2<> Nil Faire
       Inser (R1, Clé(R2))
       Suppr(R2, Clé(R2))
   FTQ ;
  Fusion ← R1 ;
Fin;
```