2LMD **18 Mars 2009**

Examen de Rattrapage ALGO1

14h50-16h20 **Amphi 2**

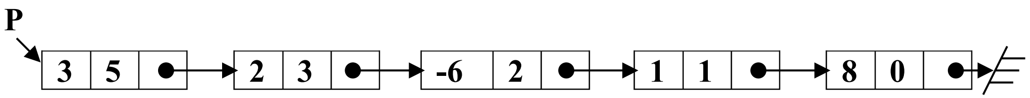
Exercice 1 (12 Pts: 1 + 2 + 3 + 3 + 3)

On convient de représenter les polynômes par des listes linéaires chaînées triées dans l'ordre décroissant des puissances où chaque maillon contient un exposant et le cœfficient correspondant.

Exemple: Le polynôme

$$P(x) = 3x^5 + 2x^3 - 6x^2 + x + 8$$

est représenté par la liste P suivante :



Ecrire les fonctions suivantes :

1. fonction **Degré(P :Pointeur(TMaillon)) :entier** qui retourne le degré du polynôme P donné.
2. fonction **PValeur(P :Pointeur(TMaillon), x : reel) :reel** qui retourne la valeur du polynôme P pour x donné. (utiliser x ^ y = x^y)
3. fonction **Somme(P1,P2 :Pointeur(TMaillon)): Pointeur(TMaillon))** qui retourne la somme des deux polynômes P1 et P2.
4. fonction **Dérivé(P:Pointeur(TMaillon)): Pointeur(TMaillon))** qui retourne le polynôme égal à la dérivée du polynôme P.
5. fonction **Intégrale(P:Pointeur(TMaillon)): Pointeur(TMaillon))** qui retourne le polynôme égal à l'intégrale du polynôme P.

N.B : Utiliser les primitives suivantes:

Aff_ValC(P,c) pour changer le coefficient d'un maillon,

Aff_ValE(P,e) pour changer l'exposant d'un maillon,

ValC(P) pour lire le coefficient d'un maillon,

ValE(P) pour lire l'exposant d'un maillon.

Exercice 2 (3 Pts: 1.5 + 1 + 0.5)

1. Dessiner le tas obtenu en insérant les éléments de la liste suivante dans l'ordre d'arrivée: 10, 5, 12, 8, 7, 1, 2, 4, 6.
2. Donner le tableau représentant le tas statique correspondant.
3. Dessiner le tas après un retrait.

Exercice 3 (5 Pts : 2 + 3)

On dit qu'un arbre de recherche binaire est équilibré si pour tout nœud de l'arbre, la différence entre la hauteur du sous-arbre gauche et du sous-arbre droit est d'au plus égale à 1.

1. Ecrite la fonction **Hauteur(R: Pointeur(TNoeud)): Entier** qui retourne la hauteur de l'arbre de recherche binaire de racine R.
2. Ecrire la fonction **Equilibré(R : Pointeur(TNoeud)) : Booleen** qui retourne vrai si l'arbre de recherche binaire de racine R est équilibré et faux sinon.

Bon courage

A.Djeffal

Corrigé type

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

1. (1 Pt)

```
fonction Degré(P :Pointeur(TMaillon)) :entier ;
Debut
    Si P=Nil alors Degré ← 0
              Sinon Degré ← ValE(P)
    FSi
Fin ;
```

2. (2 Pts)

```
fonction PValeur(P :Pointeur(TMaillon), x : reel) :reel
Var S : reel ;
Debut
    S←0 ;
    TQ P ≠ Nil Faire
        S ← S + ValC(P)*(x ^ ValE(P));
        P ← Suivant(P);
    FTQ
    PValeur ← S ;
Fin ;
```

3. (3 Pts)

```
fonction Somme(P1,P2 :Pointeur(TMaillon)): Pointeur(TMaillon)) ;
Var Tete_Sm, Q, P, P12 : Pointeur(TMaillon);
Debut

  Tete_Sm ← Nil ;   // tête de la liste somme
  Q ← Nil ;         // dernier élément de la liste somme

TQ P1 ≠ Nil et P2 ≠ Nil Faire
  Allouer (P) ; Aff_Adr(P,Nil) ;
  Si Tete_Sm=Nil alors Tete_Sm ← P ;
                 Sinon Aff_Adr(Q,P) ;
  FSi
  Q ← P;
  Si ValE(P1) = ValE(P2) alors
      Aff_ValE(P, ValE(P1)) ; Aff_ValC(P, ValC(P1) + ValC(P2)) ;
      P1 ← Suivant(P1);
      P2 ← Suivant(P2);
  Sinon
      Si ValE(P1) > ValE(P2) alors
          Aff_ValE(P, ValE(P1)) ; Aff_ValC(P, ValC(P1)) ;
          P1 ← Suivant(P1);
      Sinon
          Aff_ValE(P, ValE(P2)) ; Aff_ValC(P, ValC(P2)) ;
          P2 ← Suivant(P2);
      FSi
  FSi
FTQ ;

Si P1 ≠ alors P12 ← P1          // P12 pointe l'éventuel reste de l'une des
      Sinon P12 ← P2 ;          // deux listes
FSi

TQ P12 ≠ Nil  Faire             // recopier le reste de la liste
  Allouer (P) ; Aff_Adr(P,Nil) ;
  Si Tete_Sm=Nil alors Tete_Sm ← P ;
                 Sinon Aff_Adr(S,P) ;
  FSi
  Q ← P;
  Aff_ValE(P, ValE(P12)) ; Aff_ValC(P, ValC(P12)) ;
  P12 ← Suivant(P12);
FTQ
Somme ← Tete_Sm ;
Fin ;
```

4. (3 Pts)

```
fonction Dérivé(P:Pointeur(TMaillon)): Pointeur(TMaillon))
Var Tete_D, Q, P1: Pointeur(TMaillon);
Debut
  Tete_D ← Nil ;   // tête de la liste dérivée
  Q ← Nil ;           // dernier élément de la liste dérivé
  TQ P ≠ Nil et ValE(P) ≠ 0  Faire
    Allouer (P1) ; Aff_Adr(P1,Nil) ;
    Si Tete_D = Nil alors Tete_D ← P1 ;
                    Sinon Aff_Adr(Q,P1) ;
    FSi
    Q ← P1;
    Aff_ValC(P1, ValC(P) * ValE(P)) ;
    Aff_ValE(P1, ValE(P) - 1) ;
    P ← Suivant(P);
  FTQ ;
  Dérivé ← Tete_D ;
Fin ;
```

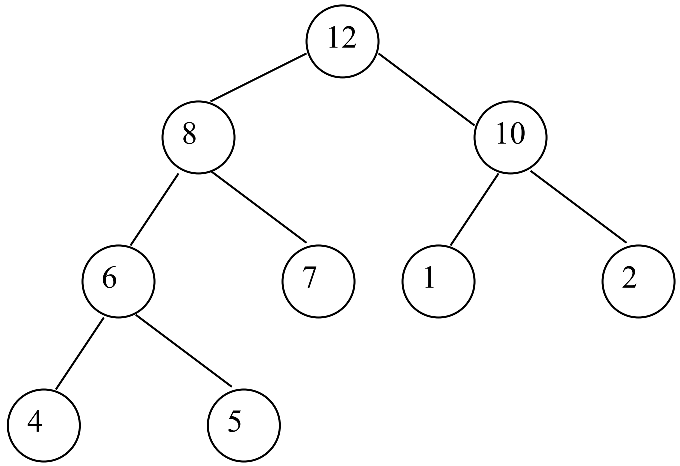
5. (3 Pts)

```
fonction Intégrale(P:Pointeur(TMaillon)): Pointeur(TMaillon))
Var Tete_Int, Q, P1: Pointeur(TMaillon);
Debut
  Tete_Int ← Nil ;   // tête de la liste intégrale
  Q ← Nil ;             // dernier élément de la liste intégrale

  TQ P ≠ Nil   Faire
    Allouer (P1) ; Aff_Adr(P1,Nil) ;
    Si Tete_D = Nil alors Tete_D ← P1 ;
                    Sinon Aff_Adr(Q,P1) ;
    FSi
    Q ← P1;
    Aff_ValC(P1, ValC(P) / (ValE(P) + 1)) ;
    Aff_ValE(P1, ValE(P) + 1) ;
    P ← Suivant(P);
  FTQ ;
   // Ajouter une constante C (supposons 1)
  Allouer (P1) ; Aff_Adr(P1,Nil) ;
  Si Tete_D = Nil alors Tete_D ← P1 ;
                  Sinon Aff_Adr(Q,P1) ;
  FSi
  Aff_ValC(P1, 1) ;
  Aff_ValE(P1, 0) ;
  Intégrale ← Tete_Int;
Fin ;
```

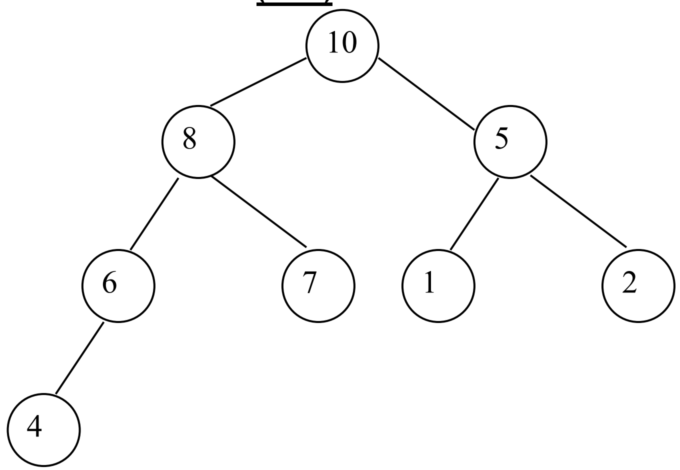
Exercice 2 (3 Pts: 1.5 + 1 + 0.5)

1. (1.5 Pt)



2. Le tas statique : 12 8 10 6 7 1 2 4 5 **(1 Pt)**

3. (0.5 Pt)



<u>Exercice 3 (5 Pts : 2 + 3)</u>

1. <u>(2 Pts)</u>

```
fonction Hauteur(R: Pointeur(TNoeud)): Entier
Debut
 Si R = Nil alors Hauteur ← 0
 Sinon Hauteur ← 1 + Max ( Hauteur (FG(R)) , Hauteur(GD(R)) )
FSi
Fin ;
```

2. <u>(3 Pts)</u>

```
fonction Equilibré(R : Pointeur(TNoeud)) : Booleen
Debut
 Si R = Nil alors Equilibré ← Vrai
 Sinon
   Equilibré ← (Abs(Hauteur (FG(R)) - Hauteur(GD(R)) ) ≤ 1) et
                          Equilibré(FG(R)) et Equilibré(FD(R)) ;
 FSi
Fin ;
```