ESI / CPI Semestre 2
Structures de données
Examen 2 // 2012 2013
Barème : 5 + 4 + 6 + (2+1+2)
Durée : 2H.

1. Utiliser un parcours "Inordre" avec pile pour ranger les adresses des feuilles d'un arbre de recherche binaire (de gauche à droite) dans un tableau. Le parcours "Inordre" est défini comme suit : pour chaque nœud n ayant comme sous arbre gauche T1 et sous arbre droit T2 on applique l'ordre T1 n T2.

2. Déterminer la branche la plus à gauche de l'arbre et la mettre dans une pile P. Une branche est l'ensemble de tous les nœuds sur le chemin de la racine vers une feuille.

3. Utiliser P pour déterminer et afficher la distance de chaque feuille vers la feuille suivante à droite. La distance désigne le nombre de nœuds traversés.

4. Utiliser le tableau construit en 1. pour chaîner les feuilles de l'arbre de gauche à droite et de droite à gauche sans l'ajout de champs "Pointeur" dans la structure du noeud. Donner alors un moyen de distinguer les nœuds feuilles de ceux non feuilles. En déduire l'algorithme de parcours des feuilles de gauche à droite.

Corrigé Algo SDD / Examen 2 / 1213

```
SOIT
   A UN ARB ;
   Tab UN VECTEUR ( 10 ) DE ARB ;
   Ind UN ENTIER ;
   In2 UNE ACTION ;
   Branche_gauche UNE ACTION ;
   Distance , Distance_feuille DES ACTIONS ;
   Afficher_tab UNE ACTION ;
   Parcours_feuilles , Chainer_feuilles DES ACTIONS ;
   SOIT Une_pile UNE PILE DE ARB ;
    DEBUT
   CREER_ARB ( A , [ 55 , 35 , 22 , 30 , 25 , 28 , 40 , 63 , 57 , 60 , 100 ] ) ;
   Ind := 0 ;
   APPEL In2 ( A ) ;     ECRIRE ( 'Feuilles de l''arbre' ) ;
   APPEL Afficher_tab ( 1 ) ;
   ECRIRE ( 'Branche gauche' ) ;     APPEL Branche_gauche ( A , Une_pile ) ;
   APPEL Distance ;
   APPEL Chainer_feuilles ;
   ECRIRE ( 'les feuilles de gauche à droite' ) ;     APPEL Parcours_feuilles ;

FIN
ACTION Afficher_tab ( I ) ;
SOIT
   I UN ENTIER ;

DEBUT
   SI I <= Ind
      ECRIRE ( INFO ( ELEMENT ( Tab [ I ] ) ) ) ;
      APPEL Afficher_tab ( I + 1 ) ;
   FSI
FIN
```

Parcours Infixe

```
ACTION In2 ( A )
SOIT
   A UN ARB ;
   SOIT Pil UNE PILE DE ARB ;
   SOIT P UN POINTEUR VERS UN ARB ;
   Possible UN BOOLEEN ;

DEBUT
   CREERPILE ( Pil ) ;
   P := A ;
   Possible := VRAI ;
   TQ Possible :
      TQ P <> NIL :
         EMPILER ( Pil , P ) ;
         P := FG ( P )
      FTQ ;
      SI NON PILEVIDE ( Pil ) :
         DEPILER ( Pil , P ) ;
         SI ( FG ( P ) = NIL ) ET ( FD ( P ) = NIL )
            Ind := Ind + 1 ;
            AFF_ELEMENT ( Tab [ Ind ] , P )
         FSI ;
         P := FD ( P )
      SINON
         Possible := FAUX
      FSI
```

```
    FTQ
FIN
ACTION Branche_gauche ( A , Pile_branche )
SOIT
    A UN ARB ;
    P UN ARB ;
    Arret UN BOOLEEN ;
    SOIT Pile_branche UNE PILE DE ARB ;

DEBUT
    CREERPILE ( Pile_branche ) ;
    P := A ;
    Arret := FAUX ;
    TQ NON Arret
        EMPILER ( Pile_branche , P ) ;
        ECRIRE ( INFO ( P ) ) ;
        TQ FG ( P ) <> NIL
            EMPILER ( Pile_branche , FG ( P ) ) ;
            ECRIRE ( INFO ( FG ( P ) ) ) ;
            P := FG ( P )
        FTQ ;
        SI FD ( P ) = NIL
            Arret := VRAI
        SINON
            P := FD ( P ) ;

        FSI
    FTQ
FIN
ACTION Distance_feuille ( F )
SOIT
    F , P , Parent DES ARB ;
    Remonte , Arret DES BOOLEENS ;
    Distance UN ENTIER ;

DEBUT
    Distance := - 1 ;
    P := F ;
    Remonte := VRAI ;
    TQ Remonte
        DEPILER ( Une_pile , Parent ) ;
        Distance := Distance + 1 ;
        SI Parent = NIL
            Remonte := FAUX
        SINON
            SI ( FG ( Parent ) = P ) ET ( FD ( Parent ) <> NIL )
                Remonte := FAUX
            SINON
                P := Parent
            FSI
        FSI
    FTQ ;
    SI Parent <> NIL
    {Important}
        EMPILER ( Une_pile , Parent ) ;
        P := FD ( Parent ) ;
        Arret := FAUX ;
        TQ NON Arret
            EMPILER ( Une_pile , P ) ;
            Distance := Distance + 1 ;
```

Branche la plus à gauche

Distance de chaque feuille vers la suivante

```
        TQ FG ( P ) <> NIL
          EMPILER ( Une_pile , FG ( P ) ) ;
          Distance := Distance + 1 ;
          P := FG ( P )
        FTQ ;
        SI FD ( P ) = NIL
          Arret := VRAI
        SINON
          P := FD ( P ) ;

        FSI
      FTQ ;
      ECRIRE ( 'feuille suivante = ' , INFO ( P ) , ' Distance=' , Distance ) ;

    FSI
FIN
ACTION Distance ;
SOIT
  I UN ENTIER ;

DEBUT
  POUR I := 1 , Ind - 1
    APPEL Distance_feuille ( ELEMENT ( Tab [ I ] ) )
  FPOUR
FIN
ACTION Chainer_feuilles ;
SOIT
  I UN ENTIER ;

DEBUT
  POUR I := 1 , Ind
    AFF_INFO ( ELEMENT ( Tab [ I ] ) , - INFO ( ELEMENT ( Tab [ I ] ) ) ) ;
    SI I > 1
      AFF_FG ( ELEMENT ( Tab [ I ] ) , ELEMENT ( Tab [ I - 1 ] ) )
    FSI ;
    SI I < Ind
      AFF_FD ( ELEMENT ( Tab [ I ] ) , ELEMENT ( Tab [ I + 1 ] ) )
    FSI ;

  FPOUR
FIN
ACTION Parcours_feuilles
SOIT
  P UN ARB ;

DEBUT
  P := ELEMENT ( Tab [ 1 ] ) ;
  TQ P <> NIL
    ECRIRE ( - INFO ( P ) ) ;
    P := FD ( P )
  FTQ ;

FIN
```

2 moyens : * soit rapatrier un bit au niveau du nœud

3/3

* soit considérer que les valeurs de l'arbre sont positives et attribuer aux feuilles des valeurs négatives (solution adoptée)