TD4- LES ARBRES BINAIRES (CORRIGÉ)

Exercices :

Soit A un arbre binaire représenté par des pointeurs (utiliser les types NŒUD et AB définis en cours).

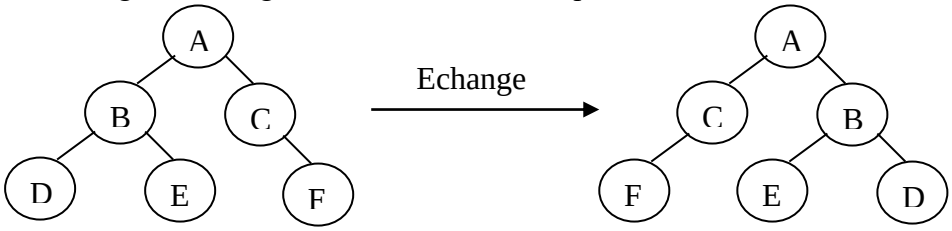
1. Ecrire une fonction récursive qui calcule le nombre de feuilles dans un arbre binaire

```
Fonction  NbFeuilles ( A : AB) : entier
Début
        Si  (A = Nil)  alors  Retourner ( 0 )
        Sinon
                Si ( A^.fg = nil et A^.fd = nil)  alors // si A est une feuille
                        Retourner ( 1 )
                Sinon
                        Retourner (NbFeuilles (A^.fg)  + NbFeuilles (A^.fd) )
                Finsi
        Finsi
Fin
```

2. Ecrire une fonction qui calcule le nombre de nœuds d'un arbre binaire.

```
Fonction  NbNoeuds ( A : AB) : entier
Début
        Si  (A = Nil)  alors  Retourner ( 0 )
        Sinon
                Retourner ( 1 + NbNoeuds (A^.fg)  + NbNoeuds (A^.fd) )
        Finsi
Fin
```

3. Ecrire une procédure récursive **Echange (A : AB)** qui étant donné un arbre binaire A permet d'échanger les fils gauches et droits de chaque nœud.




```
Procédure   Echange ( A : AB)
Var  P : ^ NŒUD
Début
        Si  (A ≠ Nil)  alors
            P ← A^.fg
            A^.fg ← A^.fd
            A^.fd ← P
            Echange ( A^.fg)
            Echange ( A^.fd)
        Finsi
Fin
```

Parcours Préfixé

Traiter (A)

4. Ecrire une fonction récursive **Egaux(A, B : AB) : booléen** qui permet de tester si deux arbres binaires A et B sont égaux.

```
Fonction   Egaux ( A, B : AB) : booléen
Début
        Si  (A ≠ Nil  et  B ≠ Nil )  // si A et B sont non vides
        alors
                Si ( A^.val = B^.val)
                 alors     Retourner ( Faux )
                Sinon
                  Retourner (Egaux (A^.fg, B^.fg)  et  Egaux (A^.fd, B^.fd) )
                Finsi
        Sinon
                Si  (A ≠ Nil  et  B ≠ Nil )  // si A et B sont vides
                alors  Retourner ( Vrai )
                Sinon
                  Retourner ( Faux )
                Finsi
Fin
```

5. Ecrire une fonction récursive **Parfait (A : AB, K : entier) : booléen** qui vérifie qu'un arbre binaire non vide de hauteur ***k*** est parfait.

```
Fonction    Parfait ( A: AB, k : entier) : booléen
Début
   Si  ( k = 1)  alors
         Retourner ( Vrai)
   Sinon
         Si  (A^.fg = Nil  ou  A^.fd = Nil )  // si A est un nœud ayant un seul fils
         alors
                  Retourner ( Faux)
         Sinon
                    Retourner (Parfait (A^.fg, k-1)  et  Parfait (A^.fd, k-1) )
         Finsi
     Finsi
Fin
```