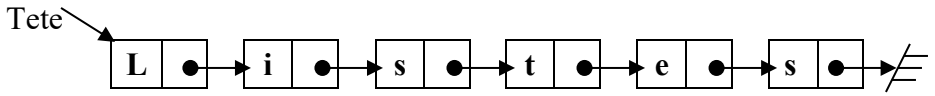
2LMD **11 Fev 2010**

Examen d'algorithmique 1

08h-09h30 **A1, A2**

Exercice 1 (10 pts: 1.5 + 1.5 + 3 + 1.5 + 2 .5)

On souhaite représenter les chaînes de caractères par des listes linéaires chaînées où chaque caractère d'une chaîne est rangé dans un maillon de la liste :
Exemple :
La chaîne de caractères "Listes" est représentée par la liste suivante :



Il est demandé d'écrire en langage algorithmique vu dans le cours les fonctions suivantes : '

1. Fonction **NbreOccur**(L :Pointeur(TMaillon), C : caractère) : entier
Qui retourne le nombre d'occurrences du caractère C dans la chaîne de caractères représentée par la liste de tête L.

2. Fonction **Concat**(L1, L2 : Pointeur(TMaillon)) : Ponteur(TMaillon) ;
Qui retourne la liste représentant le résultat de concaténation des deux chaines représentées par L1 et L2.

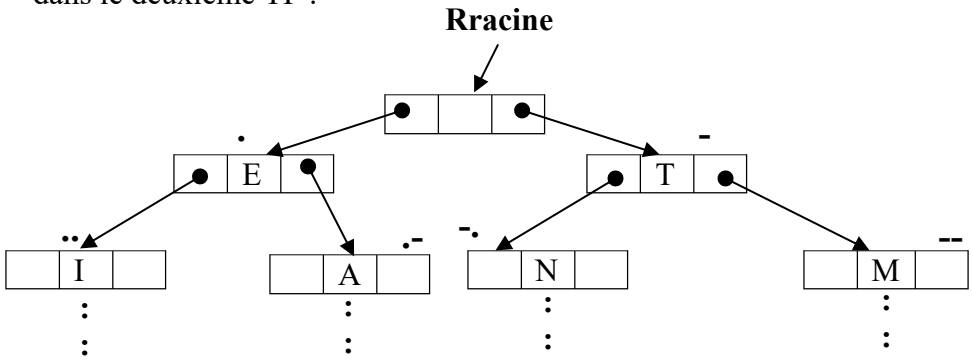
3. Fonction **Retirer**(L:Pointeur(TMaillon), C:caractère):Pointeur(TMaillon);
Qui retourne une liste égale à la liste L sans le caractère C.

4. Fonction **Position**(L:Pointeur(TMaillon), C:caractère):Pointeur(TMaillon);
Qui retourne un pointeur sur la première occurrence du caractère C dans la liste L.

5. Fonction **Remplacer**(L:Pointeur(TMaillon), C1,C2:caractère):
Pointeur(TMaillon);
Qui retourne une liste égale à L avec toutes les occurrences de C1 remplacées par C2.

Exercice 2 (10 pts : 3 + 1.5 + 1.5 + 4)

Soit l'arbre binaire de recherche représentant les codes morse vu dans le deuxième TP :



Ecrire les fonctions suivantes :

1. Fonction **NbrePréfixes** (Racine: Pointeur(TNoeud),Ch: chaîne):entier ;
Qui retourne le nombre de caractères dont le code commence par la chaîne Ch.

2. Fonction **NbrePoints** (Racine: Pointeur(TNoeud)):entier ;
Qui retourne le nombre de points dans tous les codes de l'arbre.

3. Fonction **NbreTirets** (Racine: Pointeur(TNoeud)):entier ;
Qui retourne le nombre de tirets dans tous les codes de l'arbre.

4. Procedure **Ajouter** (Racine: Pointeur(TNoeud),C:Caractère, Code:chaîne) ;
Qui ajoute le caractère C de code Code à l'arbre.

Bon courage

Corrigé type

Exercice 1 (10 pts: 1.5 + 1.5 + 3 + 1.5 + 2 .5)

1. Fonction **NbreOccur**(L :Pointeur(TMaillon), C : caractère) : entier

```
   Var P : Pointeur(TMaillon) ;
       Nb : entier ;
Debut
   P ← L ;
   Nb ← 0 ;
   TQ P≠NIL faire
      Si Valeur(P) = C alors
           Nb ← Nb + 1
      Fsi ;
      P ← Suivant(P) ;
   FTQ ;
   NbreOccur ← Nb ;
Fin ;
```

(1.5 Pt)

2. Fonction **Concat**(L1, L2 : Pointeur(TMaillon)) : Pointeur(TMaillon) ;

```
   Var P, P1, NTete, Q : Pointeur(TMaillon) ;
Debut
   P ← L1 ;
   NTete ← NIL ;
   TQ P≠NIL  faire
      Allouer(P1) ;
      Aff_Val(P1,Valeur(P));
      Aff_Adr(P1,NIL) ;
      Si NTete=NIL alors  NTete ← P1
                    Sinon Aff_Adr(Q,P1)
      Fsi
      Q ← P1 ;
      P ← Suivant(P) ;
   FTQ ;
   P ← L2 ;
   TQ P≠NIL  faire
      Allouer(P1) ;
      Aff_Val(P1,Valeur(P));
      Aff_Adr(P1,NIL) ;
      Si NTete=NIL alors  NTete ← P1
                    Sinon Aff_Adr(Q,P1)
      Fsi
      Q ← P1 ;
      P ← Suivant(P) ;
   FTQ ;
   Concat ← NTete ;
Fin ;
```

(1.5 Pt)

3. Fonction **Retirer**(L:Pointeur(TMaillon), C:caractère):Pointeur(TMaillon);

```
   Var P, P1, NTete, Q : Pointeur(TMaillon) ;
Debut
   P ← L1 ;
   NTete ← NIL ;
   TQ P≠NIL  faire
      Si Valeur(P) ≠ C alors
         Allouer(P1) ;
         Aff_Val(P1,Valeur(P));
         Aff_Adr(P1,NIL) ;
         Si NTete=NIL alors  NTete ← P1
                       Sinon Aff_Adr(Q,P1)
         Fsi
         Q ← P1 ;
      Fsi
      P ← Suivant(P) ;
   FTQ ;
   Retirer ← NTete ;
Fin ;
```

(3 Pts)

4.Fonction **Position**(L:Pointeur(TMaillon), C:caractère):
Pointeur(TMaillon);

```
Var P : Pointeur(TMaillon) ;
Debut
    P ← L ;
    TQ P≠NIL et Valeur(P)≠C faire
        P ← Suivant(P) ;
    FTQ ;
    Position ← P;
Fin ;
```

(1.5 Pt)

5. Fonction **Remplacer**(L:Pointeur(TMaillon), C1,C2:caractère):
Pointeur(TMaillon);

```
Var P, P1, NTete, Q : Pointeur(TMaillon) ;
Debut
    P ← L;
    NTete ← NIL ;
    TQ P≠NIL  faire
        Allouer(P1) ;
        Si Valeur(P) = C1 alors Aff_Val(P1,C2);
                          Sinon Aff_Val(P1,Valeur(P));
        Fsi ;
        Aff_Adr(P1,NIL) ;
        Si NTete=NIL alors  NTete ← P1
                     Sinon Aff_Adr(Q,P1)
        Fsi
        Q ← P1 ;
        P ← Suivant(P) ;
    FTQ ;
    Remplacer ← NTete ;
Fin ;
```

(2.5 Pt)

Exercice 2 (10 pts : 3 + 1.5 + 1.5 + 4)

1. Fonction **NbrePréfixes** (Racine: Pointeur(TNoeud),Ch: chaîne):entier ;

```
Var  R : Pointeur(TNoeud) ;
     Ch1 : Chaine ;
     i : entier ;
Debut
    Si Racine = NIL alors NbrePréfixes ← 0
    Sinon
        Si Ch='' alors NbrePrefixes ← Taille(Racine)
        Sinon
            Si Ch[1] = '.' alors  R ← FG(Racine)
                          Sinon R ← FD(Racine)
            Fsi
            Ch1 ← ''
            Pout i ←2 à Longueur(Ch) faire
                Ch1 ← Ch1 + Ch[i];
            FPour ;
            NbrePréfixes ← NbrePréfixes(R, Ch1) ;
        Fsi;
    Fsi
Fin ;
```

```
Fonction Taille (R : Pointeur(TNoeud)) :entier ;
Debut
    Si R = NIL alors Taille ← 0
    Sinon
        Taille ← 1 + Taille(FG(R)) + Taille(FD(R))
    Fsi
Fin ;
```

(3 Pts)

2. Fonction **NbrePoints** (Racine: Pointeur(TNoeud)):entier ;

```
Var Nb :entier ;
Debut
    Nb ← 0 ;
    NbrePoints ← NbPts(Racine, Nb) ;
Fin ;
```

```
Fonction NbPts(R : Pointeur(TNoeud), N :entier) : entier ;
Var NG, ND :entier ;
Debut
  Si R = NIL alors NbPts ← N
  Sinon
    Si FG(R) = Nil  alors   NG ← 0 ;
                    Sinon   NG ← NbPts(FG(R), N+1)
    Fsi ;
    Si FD(R) = Nil  alors   ND ← 0 ;
                    Sinon   ND ← NbPts(FD(R), N)
    Fsi ;
     NbPts ← N + NG + ND ;
  Fsi
Fin ;                                          (1.5 Pt)
```

3. Fonction **NbreTirets** (Racine: Pointeur(TNoeud)):entier ;

```
Var Nb :entier ;
Debut
  Nb ← 0 ;
  NbreTirets ← NbTrs(Racine, Nb) ;
Fin ;

Fonction NbTrs(R : Pointeur(TNoeud), N :entier) : entier ;
Var NG, ND :entier ;
Debut
  Si R = NIL alors NbPts ← N
  Sinon
    Si FG(R) = Nil  alors   NG ← 0 ;
                    Sinon   NG ← NbTrs(FG(R), N)
    Fsi ;
    Si FD(R) = Nil  alors   ND ← 0 ;
                    Sinon   ND ← NbTrs(FD(R), N+1)
    Fsi ;
    NbTrs ← N + NG + ND ;
  Fsi
Fin ;                                          (1.5 Pt)
```

4. Procedure **Ajouter** (Racine: Pointeur(TNoeud),C:Caractère, Code:chaîne) ;

```
Var
Debut
  Si Racine = NIL alors
     Allouer (Racine ) ;
     AFF_FG(Racine, NIL) ;
     AFF_FD(Racine,NIL) ;
  Fsi ;
  N ← Racine ;
  PN ← NIL ;
  i ← 1 ;
  TQ N ≠ NIL et i ≤ Longueur(Code) Faire
     PN ← N ;
     Si Code[i] = '.' Alors  N ← FG(N)
                     Sinon  N ← FD(N)
     Fsi ;
     i ← i + 1
  FTQ ;
  Si N = NIL alors
        Pour j = i -1 à Longueur(Code) Faire
           Allouer(N1) ;
           AFF_FG(N1, NIL) ;
           AFF_FD(N1, NIL) ;
           Si Code[j] = '.' Alors AFF_FG(PN, N1) ;
                          Sinon AFF_FD(PN, N1) ;
           Fsi
          PN ← N1 ;
        FPour ;
        AFF_Info(Q, C);
  Sinon
         AFF_Info(N, C) ;
  Fsi
Fin ;                                          (4 Pts)
```