CPI / S2 / Examen Structures de Données Dynamiques
ESI / 2012-2013 / Doc non autorisés / Durée : 2H
Barème : 6 + 2 + 8 + 2 + 2

Accélération de la recherche dans une liste linéaire chaînée

Partant du fait que tout nombre entier est de la forme 4x, 4x+1, 4x+2 ou 4x+3, on décide de ranger N données entières dans un tableau Tab de 4 listes ordonnées dépourvues de doubles; chaque liste correspond à une forme. *[Structure 1]*

Ecrire le module de construction.

En déduire le module de recherche d'une donnée.

Afin d'accélérer plus la recherche, on scinde (ou on éclate) chacune des 4 listes selon le critère "Dernier chiffre de la donnée inférieur à 5". On rajoute alors un autre tableau Tab2 de 4 listes.

Après éclatement, Tab[i] contient les données avec le dernier chiffre < 5 et Tab2[i] contient les données avec le dernier chiffre >=5. *[Structure 2]*

Ecrire le module d'éclatement sans création de nouveaux maillons.

En déduire le nouveau module de recherche.

Donner le nombre moyen de comparaisons pour retrouver une donnée dans une liste ordonnée, dans la structure 1et dans la structure 2.

Corrigé Examen 1 ALGOSDD / 2012 - 2013

```
SOIT
  I , N , Nombre DES ENTIERS ;
  Tab , Tab2 DES VECTEURS ( 4 ) DE LISTES ;
  Inserer , Inserer_liste , Recherche , Recherche2 DES ACTIONS ;
  Appartient UNE FONCTION ( BOOLEEN ) ;
  Dernier_chiffre UNE FONCTION ( ENTIER ) ;
  Afficher_tab , Afficher_liste , Scinder_tab , Scinder DES ACTIONS ;

DEBUT
  LIRE ( N ) ;
  POUR I := 1 , 4
    AFF_ELEMENT ( Tab [ I ] , NIL ) ;
    AFF_ELEMENT ( Tab2 [ I ] , NIL )
  FPOUR ;
  POUR I := 1 , N
    Nombre := ALEANOMBRE ( 100 ) ;
    APPEL Inserer ( Nombre )
  FPOUR ;
  ECRIRE ( 'Table 1' ) ;
  APPEL Afficher_tab ( Tab ) ;
  APPEL Recherche ( N + 1 ) ;
  APPEL Scinder ;
  ECRIRE ( 'Table 1' ) ;
  APPEL Afficher_tab ( Tab ) ;
  ECRIRE ( 'Table 2' ) ;
  APPEL Afficher_tab ( Tab2 ) ;
  APPEL Recherche2 ( N + 1 ) ;

FIN
FONCTION Dernier_chiffre ( N ) : ENTIER ;
SOIT
  N , Quotient DES ENTIERS ;

DEBUT
  Dernier_chiffre := MOD ( N , 10 )
FIN
ACTION Inserer ( N )
SOIT
  I , N DES ENTIERS ;
  L UNE LISTE ;

DEBUT
  I := MOD ( N , 4 ) ;
  L := ELEMENT ( Tab [ I + 1 ] ) ;
  APPEL Inserer_liste ( L , N ) ;
  AFF_ELEMENT ( Tab [ I + 1 ] , L )
FIN
```

```
ACTION Inserer_liste ( L , N )
SOIT
   L , P , Q , Prec DES LISTES ;
   N UN ENTIER ;
   Arret , Trouv DES BOOLEENS ;

DEBUT
   Prec := NIL ;
   Q := L ;
   Arret := FAUX ;
   Trouv := FAUX ;
   TQ ( Q <> NIL ) ET NON Arret ET NON Trouv
      SI VALEUR ( Q ) = N
         Trouv := VRAI
      SINON
         SI VALEUR ( Q ) > N
            Arret := VRAI
         SINON
            Prec := Q ;
            Q := SUIVANT ( Q )
         FSI
      FSI
   FTQ ;
   SI ( Q = NIL ) OU Arret
      ALLOUER ( P ) ;
      AFF_VAL ( P , N ) ;
      AFF_ADR ( P , NIL ) ;
      SI Prec = NIL
         L := P ;

      SINON
         AFF_ADR ( Prec , P ) ;
         AFF_ADR ( P , Q )
      FSI
   FSI
FIN
ACTION Afficher_tab ( Table )
SOIT
   Table UN VECTEUR ( 4 ) DE LISTES ;
   I UN ENTIER ;

DEBUT
   POUR I := 1 , 4
      APPEL Afficher_liste ( ELEMENT ( Table [ I ] ) ) ;
      ECRIRE ( '_____' ) ;

   FPOUR
FIN
ACTION Afficher_liste ( L )
SOIT
   P , L DES LISTES ;

DEBUT
   P := L ;
   TQ P <> NIL
```

```
    ECRIRE ( VALEUR ( P ) ) ;
    P := SUIVANT ( P )
  FTQ
FIN
ACTION Recherche ( N )
SOIT
  I , N DES ENTIERS ;

DEBUT
  I := MOD ( N , 4 ) ;
  ECRIRE ( Appartient ( ELEMENT ( Tab [ I + 1 ] ) , N ) )
FIN

ACTION Recherche2 ( N )
SOIT
  I , N DES ENTIERS ;

DEBUT
  I := MOD ( N , 4 ) ;
  SI Dernier_chiffre ( N ) < 5
    ECRIRE ( Appartient ( ELEMENT ( Tab [ I + 1 ] ) , N ) )
  SINON
    ECRIRE ( Appartient ( ELEMENT ( Tab2 [ I + 1 ] ) , N ) )
  FSI
FIN
FONCTION Appartient ( L , N ) : BOOLEEN ;
SOIT
  L , P DES LISTES ;
  N UN ENTIER ;
  Trouv UN BOOLEEN ;

DEBUT
  P := L ;
  Trouv := FAUX ;
  TQ ( P <> NIL ) ET NON Trouv
    SI VALEUR ( P ) >= N
      Trouv := VRAI
    SINON
      P := SUIVANT ( P )
    FSI
  FTQ ;
  SI Trouv
    Appartient := VALEUR ( P ) = N
  SINON
    Appartient := FAUX
  FSI
FIN
ACTION Scinder ;
SOIT
  I UN ENTIER ;
DEBUT
  POUR I := 1 , 4
    APPEL Scinder_tab ( I )
  FPOUR
FIN
```

```
ACTION Scinder_tab ( I ) ;
SOIT
   I UN ENTIER ;
   L , L2 , Prec , P DES LISTES ;
   Trouv UN BOOLEEN ;

DEBUT
   L := ELEMENT ( Tab [ I ] ) ;
   L2 := ELEMENT ( Tab2 [ I ] ) ;
   Prec := NIL ;
   TQ L <> NIL
      SI Dernier_chiffre ( VALEUR ( L ) ) >= 5
         SI L2 = NIL
            AFF_ELEMENT ( Tab2 [ I ] , L )
         SINON
            AFF_ADR ( L2 , L )
         FSI ;
         L2 := L ;
         SI Prec <> NIL
            AFF_ADR ( Prec , SUIVANT ( L ) )
         SINON
            AFF_ELEMENT ( Tab [ I ] , SUIVANT ( L ) )
         FSI ;

      SINON
         Prec := L ;

      FSI ;
      L := SUIVANT ( L ) ;

   FTQ ;
   AFF_ADR ( L2 , NIL ) ;

FIN
```