TP : Arbres binaires

Objectifs du TP Ce TP a pour but de vous faire manipuler des arbres binaires.

Matériel nécessaire Pour effectuer ce TP, il vous faut

1. l'unité `U_Element` définissant le type `ELEMENT` par

```
interface
type
   ELEMENT = CARDINAL;
```

Matériel fourni

- Les notes de cours Arbres (1), Arbres (2) et Arbres (3) ;
- une unité `U_Arbre-base.pas` [1] que vous serez amené à compléter ultérieurement ;
- une unité `U_Exemples_Arbres.pas` [2].

1 Présentation de l'unité `U_ExemplesArbres`

L'unité U_Exemples_Arbres ne contient que deux déclarations :

1. une constante qui détermine le nombre d'arbres définis par cette unité

```
const N = 50;
```

2. une variable de type tableaux d'arbres indéxé de 0 à `N-1`.

```
var exemple : array[0..N-1] of ARBRE ;
```

La partie initialisation de cette unité affecte un arbre binaire à chaque élément du tableau.

Remarque : Le code source de cette unité a été générée automatiquement par un programme (écrit en Pascal) en choisissant la taille des arbres aléatoirement dans un certain intervalle, ainsi que leur structure et contenu.

C'est un excellent exercice que d'écrire un programme qui produit le code source d'une telle unité.

2 Hauteur, taille et parcours d'arbres

Les deux premiers exercices sont des applications directes des opératons primitives ainsi que des quelques fonctions vues en TD.

2.1 Compléter l'unité `U_Arbre`

Question 1. Récupérez l'unité `U_Arbre-base.pas` et renommez-la en `U_Arbre.pas`.
Question 2. Ajoutez les fonctions `taille` et `hauteur` vues en cours dans la partie interface.

[1] empreinte sha1 de l'unité

```
a7ef14b962beb70071462b7fa732f0347e71b6ae  U_Arbre-base.pas
```

[2] empreinte sha1 de l'unité

```
2868f3d8c32c1234f95db4d4897eaff48dc07f60  U_Exemples_Arbres.pas
```

2.2 Étude d'un arbre

Question 3. Que vaut votre NIP modulo 50 ? Dans la suite on désigne par k ce nombre.
Question 4. Quelle est la taille de l'arbre numéro k de l'unité `U_Exemples_Arbres` ?
Question 5. Et sa hauteur ?
Question 6. Écrivez une fonction qui calcule le nombre de feuilles dans un arbre. Puis déterminez le nombre de feuilles de l'arbre d'indice k.
Question 7. Écrivez une procédure qui affiche à l'écran les éléments de l'arbre dans l'ordre infixé. Donnez la liste dans l'ordre infixé des élements contenus dans l'arbre numéro k.

2.3 Recherche d'arbres extremes

Question 8. Quel est la taille minimale d'un arbre du tableau `exemple` ? et la taille maximale ?
Question 9. Même question avec la hauteur.
Question 10. Même question avec le nombre de feuilles.

3 Arbres ordonnés

3.1 Préambule

Question 11. Modifiez l'unité `U_Element` en ajoutant dans la partie interface la fonction

```
// compareElements($e_1$,$e_2$) = -1 si $e_1<e_2$
//                              0 si $e_1=e_2$
//                              1 si $e_1>e_2$
function compareElements(e1,e2 : ELEMENT) : INTEGER;
```

Question 12. Ajoutez dans la partie interface de l'unité `U_Arbre` la fonction

```
// estABO($a$) = Vrai si $a$ est un ABO
//               Faux sinon
function estABO(a : ARBRE) : BOOLEAN;
```

3.2 Arbres binaires de l'unité `U_Exemples_Arbres`

Question 13. L'arbre situé à l'indice k (k étant l'entier calculé dans le premier exercice) est-il ordonné ?
Question 14. Parmi tous les arbres donnés en exemple, y en a-t-il un qui soit ordonné ?

3.3 Construction d'ABO

Question 15. Ajoutez dans la partie interface de l'unité `U_Arbre` la procédure vue en cours

```
// insererABO($e$,$a$)
// ajoute un nouvel élément $e$ en feuille de
// l'arbre $a$. L'arbre $a$ est modifié
procedure insererABO(e : ELEMENT;
                     var a : ARBRE) ;
```

Question 16. Réalisez un programme qui lit une suite de nombres entiers positifs donnés par l'utilisateur se terminant par 0 et construit un ABO en insérant successivement ces entiers sauf 0 dans un arbre.
Question 17. Testez votre programme en utilisant votre fonction `estABO`.
Question 18. Déterminez la hauteur de l'ABO obtenu avec les suites de nombres suivantes

I. 8 4 12 2 6 10 14 1 3 5 7 9 11 13 15
II. 8 12 4 2 10 14 6 5 13 15 1 7 11 9 3
III. 5 8 4 2 13 6 7 10 15 1 3 9 14 11 12
IV. 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
V. 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

Question 19. Écrivez un programme qui construit un ABO par insertion successive des éléments de l'arbre de l'unité `U_Exemples_Arbres` dont le numéro est votre NIP modulo 50 dans l'ordre infixé, et qui affiche les caractéristiques principales de l'arbre obtenu : taille, hauteur, racine, nombre de feuilles, taille des deux sous-arbres...