Université Mohammed V-Agdal
Faculté des Sciences de Rabat
Département d'Informatique

Année Universitaire 2013/2014
MODULE 14 (SMI-S4)

Travaux Dirigés de Structures de Données
[TD n°4 : Arbres]

Objectifs : *- Initier à la manipulation des arbres (représentés par pointeurs ou par tableaux) ;*
- Distinguer différents arbres binaires : de recherche et tas.

Exercice 1

On considère ***la représentation par pointeurs*** pour le TAD **`Arbre_Binaire`** défini en cours pour ***un arbre binaire*** contenant des nombres entiers. Ecrire *en langage **C*** les fonctions suivantes :

a)- La fonction **`taille`** qui calcule le nombre de nœuds dans un arbre binaire **`Ab`**.

b)- La fonction **`hauteur`** qui calcule la hauteur d'un arbre binaire **`Ab`**.

c)- La fonction **`est_feuille`** qui retourne **`1`** si un arbre binaire **`Ab`** correspond à une feuille, **`0`** sinon.

d)- La fonction **`appartient_AB`** qui retourne **`1`** si un élément **`e`** appartient à un arbre binaire **`Ab`**, **`0`** sinon.

Exercice 2

Un arbre binaire de recherche est un arbre binaire tel que pour tout nœud, les clés de tous les nœuds du sous-arbre gauche (*resp. du sous-arbre droit*) sont inférieures ou égales (*resp. supérieures*) à la clé du nœud.

a)- Dessiner l'arbre binaire de recherche obtenu par ajouts successifs aux feuilles des éléments suivants :
`10, 12, 7, 14, 16, 15, 4, 9, 8, 5, 3, 2`.

b)- Supprimer les éléments **`15, 9, 7`**, en utilisant l'algorithme vu en cours.

c)- Ecrire la série de nombres entiers qui résulte de l'arbre précédent en utilisant chacun des parcours suivants : préfixe, infixe, postfixe et en largeur (*par niveau).*

Exercice 3

On considère ***la représentation par pointeurs*** pour le TAD **`Arbre_Rech`** défini en cours pour ***un arbre binaire de recherche*** contenant des nombres entiers.

a)- Ecrire en ***C*** la fonction récursive **`affiche_infixe`** qui affiche, à l'aide d'un parcours infixe, les éléments d'un arbre binaire de recherche **`Abr`**.

b)- Proposer en ***pseudo-code*** l'algorithme **`affiche_en_largeur`** qui affiche, à l'aide d'un parcours en largeur, les éléments d'un arbre binaire de recherche **`Abr`**. (***Indication :*** *Utiliser la structure de données* **`File`**)

c)- (***Question facultative***) Ecrire en ***C*** la fonction *itérative* **`appartient_ABR`** qui retourne **`1`** si un élément **`e`** appartient à un arbre binaire de recherche **`Abr`**, **`0`** sinon.

d)- (***Question facultative***) Ecrire la fonction **`fusion_ABR`** qui fusionne deux arbres binaires de recherche **`Abr1`** et **`Abr2`** (*sans duplication des éléments*).

Exercice 4

Un tas est un arbre binaire *parfait* tel que la clé de chaque nœud est supérieure ou égale aux clés de ses fils. On considère ***la représentation par tableaux*** pour le TAD **`Tas`** défini en cours pour ***un tas*** contenant des entiers.

a)- Donner l'arbre binaire parfait correspondant au tableau **`T`** d'entiers suivants : {**`7`**, **`1`**, **`6`**, **`4`**, **`5`**, **`2`**, **`6`**, **`10`**, **`9`**, **`11`**, **`8`**}

b)- Transformer cet arbre pour qu'il vérifie la propriété du tas.

c)- (***Question facultative***) Exécuter l'algorithme du ***tri par tas*** sur le tableau **`T`**.

d)- (***Question facultative***) Ecrire en ***C*** la fonction **`appartient_Tas`** qui retourne **`1`** si un élément **`e`** appartient à un tas **`T`**, **`0`** sinon.