Univérsité Hassan II Casablanca

Ecole normale supérieure (ENS) Casablanca

Module : Structures de données en C (Info 4)

Année universitaire 2020/2021

Parcours 2éme année LEESM **Prof. Mohamed Lahby**

TD8

Les arbres binaires de recherches (ABR)

Exercice 1

Q1. Insérer les entiers suivants dans un ABR initialement vide, dans l'ordre. Détailler les modifications faites à l'arbre.

$$\text{int T[]} = \{14, 7, 88, 51, 17, 53, 3\}$$

Q2. Même question pour le tableau $\text{int T[]} = \{4, 20, 12, 2, 7, 3, 6, 0, 15, 1, 13, 14\}$

Exercice 2

Dans les deux exemples d'arbres binaires de recherche de la figure 1 :

Q1. où peut-on insérer un élément de clé 13 ?

Q2. comment peut-on supprimer l'élément de clé 14 ?

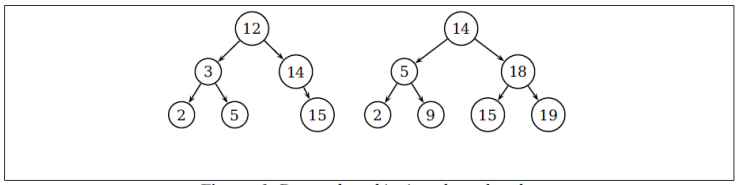

Figure 1: Deux arbres binaires de recherche

Exercice 3

On considère la déclaration suivante :

```
typedef struct noeud{
        int val;
        struct noeud* gauche;
        struct noeud* droit;
}abr;
```

Q1. Écrire une fonction récursive **void AfficherOrdcroissant(abr *A)** qui affiche l'ABR qui lui est passé en paramètre, par ordre croissant des valeurs.

Q2. Écrire une fonction récursive **void AfficherOrdDec(abr *A)** qui affiche l'ABR qui lui est passé en paramètre, par ordre décroissant des valeurs.

Q3. Écrire une fonction **int EstABR(abr *A)** qui retourne 1 si l'abre A est ABR ou 0 sinon.

Exercice 4

Q1. Écrire une fonction récursive **abr * CreerFeuille(int x)** qui permet de créer une feuille dont la valeur de la racine est x.

Q2. Écrire une fonction récursive **abr * InsertFeuille_ABR(abr * A, int e)** permet d'ajouter un noeud e dans l'arbre binaire de recherche A

Q3. Écrire une fonction **abr * Tab_To_ABR(int T[], int N)** qui permet de creer un ABR à partir d'une tableau d'entiers L.

Exercice 5

Q1. Écrire une fonction **int minimumABR(abr * A)** : qui retourne le minimum d'un ABR A.

Q2. Écrire une fonction **int maximumABR(abr * A)** : qui retourne le maximum d'un ABR A.

Exercice 6

Q1. Ecrire une fonction d'entête **int memes_valeurs(abr * a1,abr * a2)** qui teste si deux arbres binaires de recherche contiennent exactement les mêmes valeurs (mais pas nécessairement organisées de la même manière). Indication : il faut bien exploiter le fait que les arbres sont des arbres de recherche ;

Q2. Ecrire une fonction d'entête **int sous_arbre(abr * a1, abr * a2)** qui teste si l'arbre a2 est un sous-arbre de l'arbre a2, c'est-à-dire s'il existe un noeud n de a1 tel que le sous-arbre enraciné en n soit égal à a2.