NF16 - TP 4 – Les Arbres Binaires de Recherche

Introduction

Soit E= {e1, e2, …, eN} un ensemble de N entiers distincts. On souhaite représenter un tel ensemble en utilisant un arbre binaire de recherche.

Dans une représentation classique, la clé de chaque sommet correspondrait à un élément de l'ensemble E.

Mais pour utiliser moins de mémoire, nous allons représenter l'ensemble d'entiers de manière plus compacte en profitant du fait que plusieurs éléments de l'ensemble peuvent être consécutifs, formant alors un intervalle de valeurs entières qui peut être caractérisé par sa borne inférieure et sa borne supérieure. Chaque sommet de l'arbre binaire de recherche correspond alors à un intervalle de valeurs entières consécutives.

A. Structures de données

Implémenter les structures de données et types suivants :

- La structure **Sommet** (et le type correspondant **T_Sommet**) qui comporte les champs :
  - **borneInf** de type **int**
  - **borneSup** de type **int**
  - **filsGauche** de type **struct Sommet***
  - **filsDroit** de type **struct Sommet***
- Le type **T_Arbre**, qui représente un ABR, de type **T_ Sommet***

A noter :

- une valeur e_i de l'ensemble qui n'est consécutive à aucune autre valeur de l'ensemble sera alors représentée par un sommet où borneInf=borneSup=e_i
- si x et y sont deux sommets, alors soit borneSup[x]+1 < borneInf[y], soit borneSup[y]+1 < borneInf[x] (autrement x et y peuvent être fusionnés en un seul sommet z avec borneInf[z]=min(borneInf[x], borneInf[y]) et borneSup[z]=max(borneSup[x], borneSup[y])

B. Fonctions de base

1. Implémenter une fonction qui permet de créer un sommet à partir d'un entier entré en paramètre. Cette fonction renvoie un pointeur vers le sommet créé :

T_Sommet *creerSommet(int element)

2. Implémenter une fonction qui permet d'insérer un élément (c'est-à-dire un nombre entier) dans l'ABR. Cette fonction renvoie un pointeur vers la racine de l'ABR :

   T_Arbre insererElement(T_Arbre abr, int element)

   Attention : lors de l'insertion d'un nouvel élément dans l'ABR, certains sommets doivent parfois être fusionnés pour respecter la propriété définie ci-dessus, pensez-donc à gérer toutes les situations possibles

3. Implémenter une fonction qui permet de rechercher un élément dans l'ABR. Cette fonction renvoie un pointeur vers le sommet correspondant à l'intervalle contenant l'élément s'il existe dans l'ABR, elle revoie NULL sinon :

   T_Sommet *rechercherElement(T_Arbre abr, int element)

4. Implémenter une fonction qui affiche tous les sommets de l'ABR, triés par ordre croissant :

   void afficherSommets(T_Arbre abr)

   NB : l'affichage d'un sommet doit se faire sous la forme `[borneInf ;borneSup]`

5. Implémenter une fonction qui affiche tous les éléments, triés par ordre croissant :

   void afficherElements(T_Arbre abr)

6. Implémenter une fonction qui supprime un élément (c'est-à-dire un nombre entier) de l'ABR. Cette fonction renvoie un pointeur vers la racine de l'ABR :

   T_Arbre supprimerElement(T_Arbre abr, int element)

   Attention à gérer tous les cas, en particulier quand la suppression d'un élément aboutit à la suppression d'un sommet de l'ABR ou à la création d'un nouveau sommet dans l'ABR

7. Implémenter une fonction qui permet d'afficher
   - la taille (en octets) occupés par l'ABR
   - la taille (en octets) qu'aurait occupé un ABR dans la représentation classique
   - le nombre d'octets gagnés par cette représentation par intervalles

   void tailleMemoire(T_Arbre abr)

C. Programme Principal :

Programmer un menu qui propose les fonctionnalités suivantes :

1. **Insérer N éléments**
2. **Rechercher un élément**
3. **Afficher tous les sommets**
4. **Afficher tous les éléments**
5. **Supprimer un élément**
6. **Afficher la taille en mémoire**
7. **Quitter** (La mémoire allouée dynamiquement doit être libérée)

Consignes générales :

Sources

- À la fin du programme, les blocs de mémoire dynamiquement alloués doivent être proprement libérés
- L'organisation MINIMALE du projet est la suivante :
  - Fichier d'en-tête tp4.h, contenant la déclaration des structures/fonctions de base,
  - Fichier source tp4.c, contenant la définition de chaque fonction,
  - Fichier source main.c, contenant le programme principal.

Rapport

Votre rapport de quatre pages maximum contiendra :

- La liste des structures et des fonctions supplémentaires que vous avez choisi d'implémenter et les raisons de ces choix.
- Un exposé succinct de la complexité de chacune des fonctions implémentées.
- Votre rapport et vos fichiers source feront l'objet d'une remise sur Moodle dans l'espace qui sera ouvert à cet effet quelques jours suivant votre démonstration au chargé de TP (un seul rendu de devoir par binôme).