NF16 – TP4 – Arbres Binaires de Recherche

Introduction

L'objectif de ce TP est de vous faire manipuler les Arbres Binaires de Recherche (ABR). Vous allez programmer une application permettant de gérer un répertoire. Un répertoire contient un ensemble de sous-répertoires, et un attribut indiquant si le répertoire est public ou privé.
Les sous-répertoires contenus dans un répertoire seront stockés dans un ABR. Les clés de l'arbre sont les noms des sous-répertoires, et l'ordre utilisé, l'ordre lexicographique. Nous nous limiterons à la gestion d'un seul répertoire racine.

Définition des types

Définissez les types, et structures qui suivent:
1- une structure *dir* (et le type correspondant *Dir*) ayant les champs suivants

- *name* de type *char**
- *status* de type entier pour le statut public (1) ou privé (0) du répertoire
- *sub* de type pointeur sur *Node* pour l'ABR des sous-répertoires

2 - une structure node permettant de représenter les nœuds de l'ABR (et le type correspondant *Node*) ayant les champs suivants

- *dir* de type pointeur sur *Dir* (le nom du répertoire sert de clé pour l'ABR)
- *lc* de type pointeur sur *Node* pour le fils-gauche
- *rc* de type pointeur sur *Node* pour le fils-droit

Fonctions requises

3 - Écrire les fonctions suivantes permettant de créer et d'initialiser les structures

- *Dir *create_dir(char* name, int status, Node *sub)*
- *Node *create_node(Dir *dir, Node *rc, Node *lc)*

4 - Écrire les fonctions suivantes sur les ABR

- *Dir *search_dir(char* name, Node *sub)*

qui renvoie un pointeur sur le nœud de l'ABR *sub* ayant pour clé *name* si le nœud existe, *NULL* sinon (récursif).

- *int add_dir_to_sub(Dir *dir, Node **sub)*

qui ajoute dans l'ABR des sous-répertoires pointés par sub un nœud dir. Renvoie *0* si tout se passe bien, *-1* sinon.

- *void print_tree(Node *sub)*

qui affiche le nom des sous-répertoires publics (et pas le nom des privés) contenus dans *sub* par ordre lexicographique croissant.

Menu

Votre programme devra commencer en proposant à l'utilisateur de saisir le nom du répertoire racine, et son statut. Vous devrez ajouter à votre programme les fonctions que vous jugerez nécessaires pour réaliser le menu qui suit. Ce menu permet à l'utilisateur de naviguer au sein du répertoire et devra avoir cette forme :

Menu du programme

1: afficher le nom et le contenu du répertoire courant

2: créer un sous-répertoire dans le répertoire courant

3: aller dans un sous-répertoire

4: retourner à la racine

0 : quitter le programme

Indications :

- Vous pouvez utiliser deux pointeurs : un qui reste constamment sur le répertoire racine, et un autre qui se déplace en fonction des choix de l'utilisateur.
- Vous devez afficher seulement les sous-répertoires publics du répertoire courant.
- Le sous-répertoire créé est vide.
- vous pouvez vous déplacer dans un sous-répertoire privé.
- lorsque vous quittez le programme vous devez libérer toute la mémoire allouée précédemment.

Modification du programme

Ajoutez à la structure *Dir* un champ *father* de type pointeur sur *Dir*.

5 - Modifiez les fonctions nécessaires pour que à l'exécution, *father* pointe sur le répertoire père du répertoire représenté par la structure, ou sur *NULL* si la structure représente la racine du répertoire.

6 - Ajoutez l'option suivante à votre menu

- 5: aller dans le répertoire père

7 - Écrivez une fonction

- *void print_path(Dir *dir)*

qui affiche le chemin complet du répertoire pointé par dir sous la forme :

R/rep0/rep1/.../repi/

Nous supposerons que ce répertoire est de profondeur au plus *DMAX* (où *DMAX* est une constante égale à *50*).

8 - Modifiez le menu pour que l'option 1 affiche le chemin complet du répertoire courant.

Bonus

9 - Ajoutez l'option suivante à votre menu

- 6 : supprimer un sous-répertoire

qui permet à l'utilisateur de saisir le nom d'un sous-répertoire du répertoire courant qu'il souhaite supprimer, et de le supprimer.

Consignes générales

- Sources

L'organisation de votre projet sera la suivante :

- fichier d'en-tête *tp4.h* contenant les définitions de types, de structures, de constantes et les prototypes de vos fonctions.
- fichier source *tp4.c* contenant les définitions de vos fonctions.
- 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 aurez choisi d'implémenter et les raisons de ces choix.
- Un exposé succinct de la complexité de chacune des fonctions implémentées.

Votre rapport et vos trois fichiers feront l'objet d'une remise de devoir sur Moodle dans l'espace qui sera ouvert à cet effet pendant quelques jours après votre démonstration au chargé de TP.