USTHB 2010/2011
Module : Algo. License ISIL, Section B, 2ième année.
Enseignant : Nadjet Kamel

Première Epreuve de Moyenne Durée

Exercice 1 (7 points) :

On souhaite créer une liste chaînée simple d'entiers. Chaque élément de cette liste est constitué de deux parties : **valeur** et **suivant.** Le champ *val* contient une valeur entière et le champ *svt* contient l'adresse de l'élément suivant dans la liste.

1. Créer le type **Telement** qui est une structure contenant un entier (*val*) et un pointeur sur Telement (*svt*), qui contiendra l'adresse de l'élément suivant. (0.5pts)
2. Créer le type **Tliste** qui est un pointeur sur le type Telement. (0.5pts)
3. Créer la liste **ma_liste** de type Tliste. (0.5pts)

En utilisant la déclaration précédente :

4. Ecrire une fonction **cree_element** qui permet de créer un nouvel élément de la liste. Elle reçoit en paramètre la valeur N de l'élément, et renvoie l'adresse de l'élément.
   Tliste cree_element(int n) ; (2pts)
5. Écrire une fonction **ajoute_element** qui permet d'ajouter un élément de la liste à une liste. Elle reçoit en paramètre l'adresse de la liste et l'adresse de l'élément à ajouter, de manière à ce que les entiers de la liste soient triés par ordre croissant. Cette fonction renvoie un pointeur sur le premier élément de la liste ainsi modifiée.
   Tliste ajoute_element(Tliste liste, Tliste NouvElement) ; (3.5pts)

Exercice 2 (8points) :

Ecrire une fonction qui vérifie qu'une chaîne de caractère est syntaxiquement correcte du point de vue des parenthèses. Les parenthèses sont de trois types (, [et { et leur parenthèses fermantes correspondantes sont respectivement),] et }. La correction syntaxique implique qu'à chaque parenthèse ouvrante corresponde, plus loin dans la chaîne de caractères, une parenthèse fermante de même type. Le texte compris entre ces deux parenthèses doit également être correct du point de vue des parenthèses : une parenthèse ouverte doit y être refermée.

Pour cela nous utilisons une pile de caractères. Nous stockons les parenthèses ouvrantes non encore refermées dans la pile. La fonction vérifie caractère par caractère la chaîne entrée :

- Si c'est une ouvrante, elle est empilée ;
- Si c'est une fermante, l'ouvrante correspondante doit être au sommet de la pile, et elle sera dépilée.

La chaîne est syntaxiquement correcte si :

- La pile n'est jamais vide à la lecture d'une fermante ; et si

- La parenthèse fermante rencontrée est toujours de même type que la parenthèse ouvrante au sommet de la pile.
- La pile est vide lorsque la chaîne de caractère a été traitée complètement.

NB : Il faut déclarer la structure pile. Utiliser les fonctions Empiler, Depiler, Sommet, PileVide en déclarant leurs prototypes sans les définir.

Exercice 3 (5points) :

1. Définir une structure **struct Arbre** permettant de coder un nœud d'un arbre binaire contenant un entier. (1pt)
2. Écrire une fonction **affiche_arbre** qui reçoit en paramètre l'adresse de la racine d'un Arbre binaire et permet d'afficher les valeurs des nœuds de cet arbre en notation *Postfixe*. (4pts)