USTHB le 24/01/2016

Examen de Rattrapage

Exercice 1 : (5pts)

Soit T un vecteur de nombres réels de taille n (n<=1000) et soit ***x*** un nombre réel donné.

1. Ecrire une action paramétrée **Somme_2** qui vérifie s'il existe deux éléments de T dont la somme est égale à x. Donner sa complexité en justifiant votre réponse.

On suppose maintenant que le vecteur T est trié dans l'ordre croissant.

2. Ecrire une action paramétrée **Somme_2bis** qui résout la question n°1 avec une complexité O(n). Justifiez votre réponse.

Exercice 2 : (8pts)

1. Soit une liste chainée d'entiers L, écrire une action paramétrée récursive **SuppElt** qui supprime l'élément d'adresse p.
2. Soient deux listes chainées d'entiers L1 et L2, écrire une action paramétrée **Concat** qui permet de concaténer L2 à L1.
3. Soient deux listes chainées d'entiers L1 et L2, écrire une action paramétrée récursive **Inverse** qui vérifie, si L2 est l'inverse de L1.
4. Ecrire une action paramétrée **Affiche** qui affiche le contenu d'une liste chainéed'entiers L.

On considère maintenant une liste chainée ***ListeRef*** où chaque élément contient une référence (point d'entrée) vers une liste chainée d'entiers et un pointeur sur l'élément suivant.

- Donner la déclaration de ***ListeRef***

5. Ecrire le programme qui :
   - Concatène chaque liste avec sa liste inverse (si elle existe) et supprime le lien vers la liste inverse dans ***ListeRef***, jusqu'à ce que toutes les listes soient concaténées à leurs listes inverses.
   - Affiche le contenu de toutes les listes de ***ListeRef.***

Exercice 3 : (7 pts)

Un arbre binaire est un arbre où chaque nœud possède au plus deux fils : un fils gauche et un fils droit.

1. Ecrire deux actions **Minimum** et **Maximum** qui retournent la plus petite valeur et la plus grande valeur d'un arbre binaire.
2. Ecrire une action récursive **SupNoeuds** qui supprime tous les nœuds d'un arbre binaire.

Un arbre binaire de recherche est un arbre binaire possédant la propriété suivante : pour tout nœud N de l'arbre, la valeur contenue dans N est strictement supérieure à toute valeur du sous-arbre gauche de N et est strictement inférieure à toute valeur du sous-arbre droit de N.

3. Ecrire une action récursive **Verifier** qui vérifie si un arbre binaire, donné en entrée, est un arbre binaire de recherche ou non.

4. En utilisant les actions précédentes, écrire un algorithme qui vérifie pour un arbre binaire A (supposé déjà créé) :
   - S'il est un arbre binaire de recherche, il affiche tous ses éléments dans l'ordre décroissant.
   - Sinon, il supprime tous ses nœuds.

Bon courage