Fiche de TD 2

Exercice 1 : Liste chaînée mono-directionnelle LCM

Dans cet exercice, nous allons représenter les vecteurs de réels par des LCM.

typedef struct el { float val ; struct el *suiv ; } Element ;
typedef Element * Vecteur ;

1. Ecrire une fonction *Affichage* d'un vecteur représenté par une LCM. Ecrire une autre fonction d'affichage à l'envers. Ecrire la programme principal pour tester ces deux fonctions.

2. Ecrire une fonction récursive *float prodscalaire (Vecteur x, Vecteur y)* qui calcule le produit scalaire des deux vecteurs x et y. Le produit scalaire de deux vecteurs est défini par ProdScalaire(x, y) = $\sum$ x(i)*y(i) pour tous les i entre 1 et n, n est le nombre d'éléments des deux vecteurs. Lorsque les deux vecteurs ont des longueurs différentes la fonction affiche un message d'erreur.

4. Donner la version itérative de *prodscalaire*.

5. Ecrire une fonction récursive *vecteur MaxVect(Vecteur x, Vecteur y)* qui crée un vecteur contenant le maximum des deux vecteurs x et y élément par élément.

Si z = MaxVect (x, y) alors z(i) = max (x(i), y(i)) si x(i) et y(i) existent

Si x(i) (resp. y(i)) n'existe pas alors z(i) = y(i) (resp. x(i)).

X =	4	51	20	3	10	2	4	55	22	30	10	1
Y=	1	51	1	12	2	0	22					
Z = MaxVect =	4	51	20	12	10	2	22	55	22	30	10	1

6. On dispose de deux listes chaînées mono-directionnellles L1 et L2 de réels triés par ordre croissant. Ces deux listes sont connues par les deux pointeurs *p1* et *p2* de type ***Pnœud.***

6.1. Déclarer le type ***Pnoeud***.

6.2. Ecrire la fonction récursive fusion qui permet de fusionner les deux listes l1 et l2 afin d'obtenir une liste contenant l'ensemble des éléments dans l'entête est :

Pnoeud fusion (Pnœud p1, Pnœud p2)

N.B. : après fusion, les 2 listes L1 et L2 ne sont pas modifiées

Exercice 2 facultatif:

Soit la liste chaînée de noms définie par les types suivants :

typedef struct name {char nom [20] ; struct name *suiv ; } Node;
typedef Node * Lcn ;

1. Ecrire une fonction qui saisit une liste chaînée de noms à l'endroit ou à l'envers (cf TD 3). ***Lc saisie (int nb)***
2. Ecrire une fonction récursive ***max*** qui calcule la plus grande valeur de la liste chaînée.
3. Ecrire le programme principal qui teste ces deux fonctions : saisir une liste puis rechercher et afficher le max de cette liste.
4. Ecrire une fonction récursive ***supprime*** qui permet de supprimer de la liste chaînée toutes les valeurs supérieures à une valeur maximale donnée comme paramètre.
5. Compléter le programme principal qui teste ces deux fonctions .
6. Ecrire une fonction ***inverse*** qui inverse la liste chaînée (le premier élément devient le dernier, le deuxième élément devient l'avant dernier, ... etc.).