EPREUVE D'INFORMATIQUE SEPTEMBRE 2006
Durée 3H 00

Problème 1 (programmation en C)

Le but de cet exercice est d'implémenter les polynômes à cœfficients entiers sous la forme de lis tes chaînées en regardant un polynôme comme la liste de ses monômes non nuls, rangés d ans l'ordre décroissant de leurs degrés.
Pour cela, on définit un type **monome** à l'aide d'une structure :

```
struct  Struct Monome
{

      int coef ;
      int deg ;
      struct  Struct Monome* suivant ;


}
```

Typedef StructMonome*suivant ;

Un polynôme sera vu comme un pointeur sur son monomer de plus haut degré (le polynôme nul sera représenté par le pointeur NULL).D'ou la définition du **type polynome** :
typedef monome*polynome ;
Ainsi, par exemple, le polynôme P = - X à la puissance 12 + 5 X à la puissance 3 + 3
sera représenté par la liste chaînée :

(Schéma à faire à la main)

a) Ecrire une fonction **CreerMonome** qui prend en argument deux entiers a et n, qui crée un monome de coefficient de valeur a, de degré n et dont le champ suivant vaut NULL et qui retourne un pointeur sur ce monome.
b) Ecrire une fonction **Deriver** qui prend en argument un polynôme et retourne le polynôme dérivé.
c) Ecrire une fonction **Somme** qui prend en argument deux polynômes **P** et **Q**, et qui retourne le polynome somme de **P** et Q.
d) Ecrire une fonction **ProduitMonome** qui prend en argument deux entiers a et n et un polynome P et qui retourne le polynome produit du monome et de p.
e) Ecrire une fonction **Produit** qui prend en argument deux polynome P et Q, et qui retourne le polynome produit de P et Q.

Remarque : si P <>0(P non nul), il s'écrit ; deg (R) < n et on a :

P x Q = (a fois X à la puissance n + R) x Q = a fois X à la puissance n fois Q + R x Q

Problème II (Programmation en c)

On voudrait présenter un module générique curliste traitant des listes avec curseur. Un curseur désigne une position entre deux éléments d'une liste. Les fonctionna lités de ce module sont :

- la création et la destruction d'une liste avec curseur.
- La gestion du déplacement du curseur (aller au début ou à la fin de la liste, avancer ou reculer d'un cran dans la liste) ;
- L'insertion d'un élément devant le curseur et la suppression de l'élément suivant le curseur ;
- De donner la valeur de l'élément suivant le curseur ;
- De donner les renseignements sur la position du curseur (être au début, être à la fin) ;
- De donner le cardinal de la liste.

1- En appliquant le principe de masquage de l'implémentation, définir l'interface du module curliste.
2- En supposant que le programme **listing.c** fasse appel au module curliste, écrire**.** son module **making file**. On souhaite implémenter les curlistes à l'aide de liste circulaire doublement chaînée. Au départ, la liste sera composée d'un élément initial ayant pour successeur le début de la liste (initialement lui-même) et pour prédécesseur la fin de la liste (initialement lui-même).
3- Après avoir dessiné la structure de données implémentant une curliste à 4 éléments (ne pas oublier la curseur),écrire les déclarations des types du fichier source du module curliste.
4- Ecrire le code des fonctions créations et de destruction d'une curliste.
5- Ecrire le code de la fonction d'insertion d'un élément devant le curseur.
6- Ecrire le code de la fonction de suppression de l'élément suivant le curseur.
7- On désire travailler avec plusieurs curseurs. Préciser les algorithmes de suppression d'un élément et de destruction d'une curliste.