TP12 : Listes doublement chaînées et polynômes

Programmation en C (LC4)

Semaine du avril 2008

Nous définissons

```
struct monome_elem{
  int degre;
  double coef;
};
typedef struct monome_elem * monome;
```

La structure `struct monome_elem` sera utilisée pour représenter un terme d'un polynôme. Un polynôme sera codé avec une liste doublement chaînée. Le type `polynome` est le suivant :

```
struct elem{
  struct elem * suivant;
  struct elem * precedent;
  monome pval;
};
typedef struct elem * polynome;
```

On dit que le liste représentant un polynome est en forme réduite si et seulement si :
– les monômes apparaissent dans la liste par ordre croisssant de degré et
– la liste ne contient que des monômes avec des coefficients différents de 0
Dans tous les exercices qui suivent, on suppose que tous les polynômes passés en paramètres sont sous la forme de liste réduite. Les polynômes retournées par ces fonctions doivent aussi être sous la forme de liste réduite.

Exercice 1 Ecrire la fonction : `polynome add(polynome p, polynome q)` qui retourne la somme des polynômes `p` et `q`. Les listes de `p` et `q` ne doivent pas être modifiées.

Exercice 2 En utilisant la fonction `add`, écrire la fonction : `polynome add_poly(polynome p,...)` à nombre variable d'arguments, tous de type `polynome`, qui calcule et retourne la somme de tous ces polynômes. On suppose que le pointeur `NULL` marque la fin de la liste des arguments.

Exercice 3 Ecrire la fonction : `polynome tab_to_poly(int n,double tab[])` qui prend en argument un tableau `tab` de `n` éléments et construit un `polynome` tel que pour tout indice `i`, `tab[i]` est le coefficient du monôme x^i.

Exercice 4 Ecrire la fonction : `double * poly_to_tab(polynome p)` qui à partir d'un polynôme `p` construit un tableau de ses coefficients et retourne un pointeur vers le premier élément de ce tableau.

Exercice 5 Ecrire la fonction : `polynome supprimer_ieme(polynome p, int i)` qui supprime le monôme de degré `i` du polinôme `p`.

Exercice 6 Ecrire la fonction : `void ecrire(polynome p,char * chemin)` qui sauvegarde le polynôme `p` dans un fichier de chemin `chemin`.
On peut sauvegarder un polynôme de deux façons différentes :
- Dans la première méthode, on écrit dans le fichier, une après l'autre, toutes les structures `struct monome_elem` où sont stockées les monômes (inutile d'écrires les pointeurs dans un fichier).
- Dans la deuxième méthode, on transforme un polynôme en tabeau de coefficients et ensuite on écrit dans le fichier la taille du tableau suivit par les éléments du tableau.

Choissisez la méthode que vous préférez en utilisant `fwrite`.

Exercice 7 Ecrire la fonction : `polynome lire(char * chemin)` qui lit un fichier de sauvegarde construit avec le fonction `ecrire` et construit et retourne le polynôme sauvegardé. Le format de fichier dépend de celui que vous avez choisit pour le fonction `ecrire`. Vous utiliserez `fread`.

Exercice 8 Ecrire la fonction : `polynome add_monome(polynome p, monome m)` qui retourne la somme du polynôme `p` et du monôme `m`. La listes de `p` doit donc être modifiée.

Exercice 9 Ecrire la fonction : `polynome derivee(polynome p)` qui retourne la dérivée du polynôme `p`.