Travaux Pratiques d'algorithmique n°7

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Listes chaînées triées

Le but de ce TP est de définir les structures et fonctions permettant la manipulation de polynômes à coefficients entiers. On programmera notamment les fonctions suivantes :

- addition et soustraction ;
- évaluation en un point ;
- dérivation;
- multiplication;
- division.

▸ **Exercice 1.** *Définir un type* `Polynome` *permettant de manipuler les polynômes (à coefficients entiers) par listes chaînées.*

- *Écrire une fonction* `Polynome * AllouerMonome(void)`*, qui réserve de la mémoire pour un monôme et renvoie l'adresse de la cellule ainsi réservée.*
- *Écrire une fonction* `Polynome * CreerMonome(int coeff, unsigned int degre)`*, qui crée une cellule contenant le monôme passé en paramètre.*

▸ **Exercice 2.** ***Afin de faciliter la recherche d'un monôme dans un polynôme, les listes chaînées seront triées par ordre de degré croissant.***

- *Écrire une fonction* `void DetruirePolynome(Polynome *p)` *qui désalloue l'espace mémoire réservé pour le polynôme* `p` *passé en argument.*
- *Écrire une fonction récursive* `Polynome * SupprimerMonome(Polynome *p, unsigned int deg)` *qui, si il existe, supprime le monôme de degré* `deg` *du polynôme* `p`*.*
- *Écrire une fonction récursive* `Polynome * AjouterMonome(Polynome *p, int coeff, unsigned int deg)` *qui crée (ou additionne s'il existe) un monôme à un polynôme. On prendra soin de supprimer l'éventuel monôme à coefficient nul ainsi créé.*

▸ **Exercice 3.**

- *Écrire une fonction* `Polynome * SaisirPolynome()` *qui crée la liste chaînée d'un polynôme dont on demandera les termes à l'utilisateur. Il est conseillé d'utiliser les fonctions précédentes.*
- *Écrire les fonctions* `void AfficherPolynomeCroissant(Polynome *p)` *et* `void AfficherPolynomeDecroissant(Polynome *p)` *qui affichent un polynôme par ordre de degrés croissants et décroissants.*

▸ **Exercice 4.** *Écrire les fonctions suivantes :*

- `Polynome * AdditionPolynome(Polynome *p, Polynome *q)`*, qui construit de façon récursive la somme des deux polynômes passés en paramètres ;*
- `double EvalPolynome(Polynome *p, double r)`*, qui renvoie la valeur d'un polynôme en un point.*

▸ **Exercice 5.** *Écrire une fonction* `Polynome * DeriverPolynome(Polynome *p)`*, qui construit le polynôme dérivé du polynôme passé en argument.*

▸ **Exercice 6.** *Écrire la multiplication de deux polynômes.*

▸ **Exercice 7.** ***(facultatif)*** *Écrire la division euclidienne (ou division par les puissances décroissantes) de deux polynômes.*