Travaux Pratiques d'algorithmique n°5

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Listes chaînées

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

- addition et soustraction ;
- évalution en un point ;
- dérivation, intégration ;
- multiplication, division euclidienne.

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

- *Ecrire la fonction* `Polynome * AllouerMonome(void)`*, qui réserve de la mémoire pour un monôme et renvoie l'adresse de la cellule ainsi réservée.*
- *Ecrire la 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.***

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

▸ **Exercice 3.**

- *Ecrire la fonction* `Polynome * SaisirPolynome(unsigned int degmax)` *qui crée la liste chaînée d'un polynôme de degré précisé en paramètre. Il est conseillé d'utiliser les fonctions précédentes.*
- *Ecrire les fonctions* `void AfficherPolynomeCroissant(Polynome *p)` *et* `void AfficherPolynomeDecroissant(Polynome *p)` *qui affichent un polynôme par ordre de degrés croissants et décroissants.*

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

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

▸ **Exercice 5.** *Ecrire les fonctions suivantes :*

- `Polynome * Derivation(Polynome *p)`, *qui construit le polynôme dérivé du polynôme passé en paramètre ;*
- `Polynome * Primitive(Polynome *p)`, *qui construit le polynôme primitive du polynôme passé en paramètre.*

Le type des coefficients des polynômes vous semble-t-il encore bien adapté ? que suggérez-vous ?

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

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