TD et TP n°3

Autour des Polynomes

Dans une séance précédente (équations bi-carrées) le polynome avait un degré connu d'avance, et pouvait donc être représenté par un objet comportant trois variables d'instances pour conserver les trois coefficients. Dans ces 2 séances de TD et TP nous allons explorer deux façons de représenter et manipuler des polynomes de degré non connus à priori.

Exercice 1 *(à faire principalement en TP)*

Un polynome sera représenté par le tableau de ses coefficients. C'est à dire qu'un polynome de degré n sera identifié un tableau `coeffs` de taille $n+1$, où pour tout indice i, `coeffs[i]` sera le coefficient du monome x^i

1. Ecrire une classe publique *Polynome*, avec comme propriété privée `coeffs` de type *double*[].
2. Ajouter un constructeur qui prendra en argument un tableau (attention à ce que vous faites de ce tableau).
3. Ecrivez trois méthodes `String toString(); int getDegre(); double getCoeff(int i);` qui donnent respectivement un moyen d'afficher un polynome, le degré du polynome, et le coéfficient du monome de degré i. (évitez l'erreur classique sur `getDegre`)
4. Écrire une méthode `void multParConst(double c)` qui multiplie le polynome par une constante
5. Écrire une méthode `void multParMonome(int i)` qui multiplie le polynome par x^i
6. Écrire une première méthode `double eval(double x)` qui évalue le polynome en un point x. (vous pouvez regardez la documentation de la classe Math, ou réécrire à la main une fonction puissance sur les doubles.)
7. Écrire une méthode `Polynome additionner(Polynome P)` qui prend en argument un autre polynome et en retourne un troisième qui vaut la somme des deux précédents. Quel est le degré maximal que l'on puisse attendre pour la somme de deux polynomes de degrés n et m ? Peut-il être de degré inférieur ?
8. Idem pour la soustraction et la multiplication.
9. *Retour sur l'évaluation en un point*

   Lorsque l'on manipule des flottants en informatique, la multiplication est une opération assez coûteuse (elle prend nettement plus de temps qu'une addition par exemple). Il est donc bon de limiter au maximum le nombre de multiplication dans un programme.

   L'idée est de constater que le polynome $a*x^3+b*x^2+c*x+d$ peut s'écrire $((a*x+b)*x+c)*x+d$

   On peut bien entendu généraliser à un polynome de degré quelconque. En déduire une méthode `double evalHorner(double x);`

Exercice 2 *Polynomes creux* - à faire principalement en TD

Il arrive très souvent que les polynomes que l'on manipule soient creux, c'est à dire que la grande majorité des coefficients sont nuls. Par exemple, $3 * x^{512} + 51 * x^{42} + 8 * x$ est un tel polynome creux. Dans une représentation semblable à celle du premier exercice, ce polynome contiendrait un tableau de taille 513 pour seulement trois coefficients !

Une solution est de ne stocker que les coefficients non nuls. Pour cela, vous pouvez utiliser la classe suivante

```
public class IndiceCoeff{
 public int indice;
 public double coeff;

 public IndiceCoeff(int i; double c){
   indice = i;
   coeff = c;
 }
}
```

1. Écrivez une classe `PolynomeCreux` avec une propriété privée `allIndiceCoeff` de type tableau de `IndiceCoeff`.
2. Ecrivez un constructeur qui prenne en argument un tableau de `IndiceCoeff` quelconque. Précisez clairement la facon dont votre constructeur traitera l'information fournie par cet argument. Discutez de la nécessité ou non de trier ces `IndiceCoeff`
3. Écrivez une méthode `String toString();` qui permetra d'afficher un polynome
4. Écrivez une méthode `int getDegre();`, qui donne le degré du polynome
5. Écrivez une méthode `void multParConst(double c)`, qui multiplie le polynome par une constante
6. Écrivez une méthode `void multParMonome(int i)` qui multiplie le polynome par x^i
7. Écrivez une méthode `double getCoeff(int i)`, qui donne le coefficient du monome i (s'il existe ...) pouvez vous écrire une recherche dichotomique
8. Ecrire une méthode `void normalise()` qui supprime toutes les `IndiceCoeff` dont le coefficient est nul. Après un appel à normalise, le tableau `allIndiceCoeff` sera de taille minimale. Que dire du polynome constant égal à 0 ?
9. Ecrire une méthode `addition` qui additionne un polynome à une autre, on devra obtenir un polynome normalisé
10. Écrire la multiplication de deux polynomes creux.