Premières associations, premières collections

On souhaite développer de quoi représenter les polynômes à une seule variable (que nous nommerons x). On rappelle qu'un polynôme est une somme de monômes, et qu'un monôme est de la forme cx^d, où c est le coefficient du monôme, et d est le degré du monôme. Le degré est un entier positif ou nul. Nous nous limiterons aux coefficients entiers.

- $3x^5$ est un monôme de coefficient 3 et de degré 5
- $4 = 4x^0$ est un monôme de coefficient 4 et de degré 0
- $5x = 5x^1$ est un monôme de coefficient 5 et de degré 1
- $4 + 5x + x^5$ est un polynôme de degré maximal 5.

Exercice 1 *Une classe Monome*

Nous allons tout d'abord implémenter une classe Monome, en respectant le diagramme de classes donné à la figure 1.

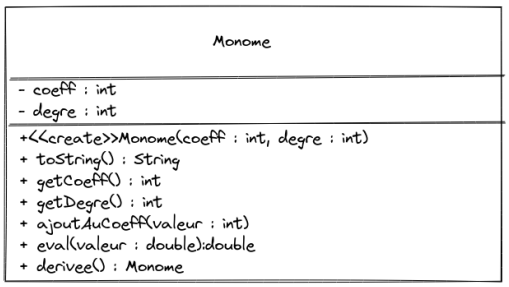

FIGURE 1 – Une classe Monome

- la méthode `toString` retournera "+3x^5" pour le monôme $3x^5$, "+4" pour le monôme $+4x^0$ et "-5x" pour le monôme $-5x^1$.
- la méthode `ajoutAuCoeff` permet d'ajouter au coefficient du monôme la valeur prise en paramètre.
- la méthode `eval` permet d'évaluer le monôme pour une valeur de x prise en paramètre. Par exemple, le monôme $3x^5$ évalué pour la valeur 2 donnera $3 \times 2^5 = 96$.
- la méthode `derivee` calcule et retourne le monome dérivé du monome receveur. On rappelle que la dérivée du monôme cx^d est cdx^{d-1}.

Vous pouvez avoir besoin des éléments suivants :

- puissance : pour effectuer le calcul de a^b, Java prévoit dans la classe `Math` une méthode de classe (static) pow (voir documentation de l'API).
- conversion d'un entier en chaîne de caractères : plusieurs solutions sont envisageables, parmi lesquelles la méthode de classe `valueOf` de la classe `String` pouvant prendre en paramètre un entier, ou bien la méthode de classe `toString` de la classe `Integer` qui prend en paramètre un int, ou encore la méthode `toString` de la classe `Integer` (méthode d'instance, non paramétrée).

Exercice 2 *Une première classe pour représenter des polynômes sous forme de liste de coefficients*

Une première représentation des polynômes que nous allons étudier ici est assez peu corrélée à la classe `Monome` que nous venons de voir. Nous allons représenter un polynôme comme la liste des coefficients de ses monômes, donc une liste d'entiers. Ainsi le polynôme $4 + 5x + x^5$ peut être réécrit en $4x^0 + 5x^1 + 0x^2 + 0x^3 + 0x^4 + x^5$ et être représenté ainsi :

0	1	2	3	4	5
4	5	0	0	0	1

Le monôme de degré i est rangé dans la "case" i de la liste. On voit ici que l'on va utiliser une structure de données proche des tableaux, puisque la position dans la liste représente ici l'indice du monôme.
La taille de la liste est au moins égale au degré maximal du polynôme (degré du monôme de plus haut degré) augmenté de 1, mais peut être arbitrairement plus grande, en rajoutant à la fin de la liste des 0, indiquant des monômes nuls de degrés supérieurs, comme dans l'exemple ci-dessous.

0	1	2	3	4	5	6	7
4	5	0	0	0	1	0	0

Implémenter cette première version des polynômes dans une classe `PolynomeDense` (dense au sens où tous les monômes, même nuls, sont représentés).
On respectera le diagramme de classe de la figure 2.

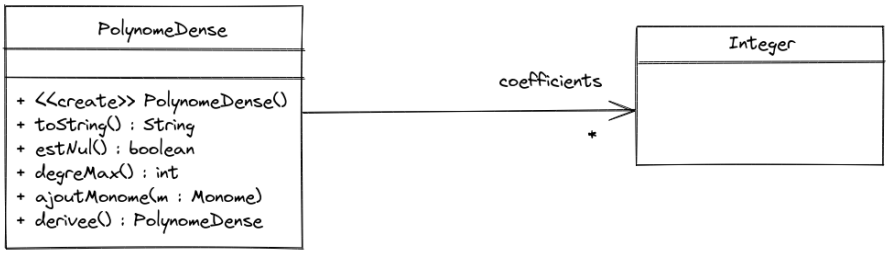

FIGURE 2 – Classe PolynomeDense

— la méthode `toString` retournera une représentation sous forme de chaîne du polynôme, dans laquelle les monômes nuls n'apparaissent pas.
— la méthode `estNul` retourne vrai si et seulement si le polynôme est nul (ne contient aucun monôme ou que des monômes nuls)
— la méthode `degreMax` retourne le degré du monôme de plus haut degré du polynôme.
— la méthode `ajoutMonome` permet d'ajouter un monôme à un polynôme. On notera au passage que suite à un ajout, un monôme peut devenir nul (si on ajoute $-3x^2$ à $3x^2$ par exemple). Mais comme notre représentation inclut des monômes nuls, ce n'est pas gênant.
— la méthode `derivee` calcule et retourne la dérivée du polynôme receveur. On rappelle que la dérivée de la somme est la somme des dérivées.

Exercice 3 *Une autre classe pour les polynômes, avec représentation sous forme de liste de monômes non nuls*

Dans la représentation précédente, on représentait tous les monômes, même les monômes nuls. Nous allons utiliser ici une autre représentation sous forme de liste de monômes non nuls, en les triant par degré croissant. On implémentera cette deuxième représentation dans une classe `PolynomeCreux`, en respectant le diagramme de classes de la figure 3.

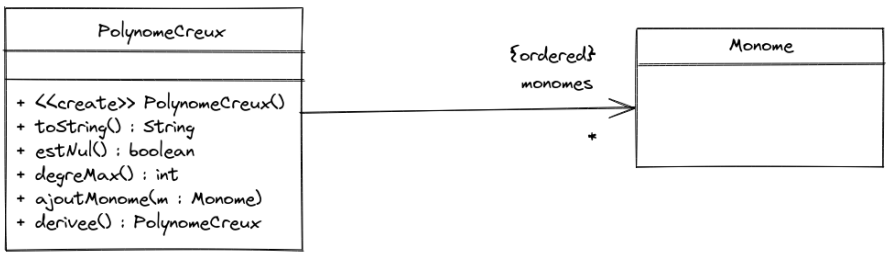

FIGURE 3 – Classe PolynomeCreux

— les méthodes sont fonctionnellement les mêmes que pour les polynômes denses.
— on fera attention au fait que cette représentation ne contient que des monômes non nuls. Quand l'ajout d'un monôme rend nul un monôme, ce monôme nul devra être supprimé du polynôme.

Exercice 4 *Des polynômes creux aux polynômes denses, et réciproquement, diagramme d'objets, références de monômes*

— Ecrire dans les deux classes de polynôme une méthode qui retourne le même polynôme sous forme de son autre représentation.
— Proposez pour le polynôme $3x + 5x^5$ deux diagrammes d'objets, l'un pour sa représentation comme instance de *PolynomeCreux*, l'autre comme instance de *PolynomeDense*.
— Imaginons le programme suivant :

</> **Programme 1 :** *Références de Monome* </>

```
Monome m1=new Monome(3,1);
Monome m2=new Monome(5,5);
PolynomeCreux p=new PolynomeCreux();
p.ajoutMonome(m1);
p.ajoutMonome(m2);
m1.ajoutAuCoeff(3); // cette instruction a t-elle un effet de bord sur p ?
```

1. La dernière instruction a t-elle dans votre implémentation un effet de bord sur p ?
2. Si oui est-ce souhaitable ? (notamment, si p était un polynôme dense, aurait-on un effet de bord ?)
3. Proposez une solution pour éviter l'effet de bord.
4. Deux instances de `PolynomesCreux` peuvent-elles maintenant avoir en commun une référence d'une instance de `Monome` ?
5. Précisez l'association de la figure 3.

Exercice 5 *Exercice supplémentaire (non fait en séance) : multiplication de polynômes*

— Mettez en place dans la classe `Monome` une méthode de classe qui calcule et retourne le produit de deux monômes (le produit de deux monômes est un monôme).

— Mettez en place dans la classe `PolynomeCreux` une méthode qui calcule et retourne le produit du polynôme receveur avec un monôme (le produit d'un polynôme avec un monôme est un polynôme). On rappelle que si on veut multiplier le polynôme P par le monôme cx^d, en posant :

$$P = \sum_{i=0}^{k} c_i x^i$$

on a :

$$P \times cx^d = cx^d \times \sum_{i=0}^{k} c_i x^i = \sum_{i=0}^{k} c \times c_i \times x^{i+d}$$

— Mettez en place dans la classe `PolynomeCreux` une méthode qui calcule et retourne la somme du polynôme receveur avec un polynôme (la somme d'un polynôme avec un polynôme est un polynôme).

— Mettez en place dans la classe `PolynomeCreux` une méthode qui calcule et retourne le produit du polynôme receveur avec un polynôme (le produit d'un polynôme avec un polynôme est un polynôme). On rappelle que si on veut multiplier le polynôme P_1 par le polynôme P_2, en posant :

$$P_2 = \sum_{i=0}^{k} c_i x^i$$

on a :

$$P_1 \times P_2 = P_1 \times \sum_{i=0}^{k} c_i x^i = \sum_{i=0}^{k} P_1 \times c_i x^i$$

— Reprenez les trois questions précédentes avec les polynômes denses.