TP 5:Chaines de caracteres et tableaux(suite)

Informatique Fondamentale (IF1)

04 novembre 2005

Exercice 1: *Indentation*

On veut ecrire un programme qui lit des lignes de programme en Java et qui les indente correctement. Il s'agit de reecrire chaque ligne en s'assurant qu'elles commencent par le bon nombre de caracteres "espace", ce nombre etant lie au nombre d'accolades ouvertes. La bonne mesure au niveau de chaque ligne est le nombre d'accolades ouvertes depuis le debut du programme et qui n'ont pas encore ete fermees.

On aura recours aux fonctions de la classe `Deug` qui permettent de manipuler les chanes de caracteres (comme `Deug.length`, `Deug.charAt` et `Deug.subString`).

q

1. Ecrire une fonction `accolades` qui prend en argument une chane de caracteres et renvoie le nombre d'accolades ouvertes par cette chane, c'est-a-dire le nombre d'occurrences de `{` moins le nombre d'occurrences de `}`.
2. Ecrire une fonction qui prend en argument une chane de caracteres et renvoie la meme chane sans espaces au debut.
3. Par convention, si n accolades sont ouvertes (et pas refermees) avant une ligne donnee, on veut qu'il y ait $4n$ espaces au debut de cette ligne. Ecrire une fonction qui prend en argument un entier n et une chane de caracteres et revoie la meme chane avec exactement $4n$ espaces au debut.
4. Ecrire un programme qui lit les lignes d'un programme avec la fonction `readLine` et ecrit les memes lignes avec le bon nombre d'espaces au debut.

Note : si on lance ce programme avec la commande `java Indentation`, le programme va attendre que l'utilisateur tape les lignes au clavier. Si on le lance avec la commande `java Indentation j Toto.java`, il va lire les lignes dans le fichier `Toto.java` et afficher le resultat.

1 Polynomes

La serie d'exercices qui suit a pour but final d'adapter l'algorithme d'Euclide de l'exercice ?? afin de calculer le pgcd de deux polynomes. Au cours de ces exercices, on representera les polynomes a coefficients reels a l'aide de tableaux de doubles. Ainsi, $P = \sum_{i=0}^n a_i X^i$ sera represente par un tableau t tel que $t[i] = a_i$ pour tout i entre 0 et n . Remarquons qu'il y a plusieurs representations possibles pour un meme polynome : on peut toujours elargir une representation sous forme de tableau par un autre tableau plus long et complete par des zeros. Par exemple, le tableau

-3	0	1
----	---	---

 et le tableau

-3	0	1	0	0
----	---	---	---	---

 representent tous les deux le polynome $X^2 - 3$.

Il en résulte en particulier qu'on ne peut pas déduire directement le degré d'un polynôme à partir de la taille d'un tableau qui le représente.

Exercice 2: *Fonctions utilitaires*

1. Rappelons que sur les nombres à virgule flottante (`double`, `float`), il n'y a pas de test d'égalité fiable. Commencer par écrire une méthode `egalite` prenant deux arguments de type `double` et qui retourne `true` lorsqu'ils sont proches à une constante ϵ près (avec par exemple $\epsilon = 0.000001$).
2. Écrire une méthode `litPolynome` qui demande à l'utilisateur de taper les coefficients d'un polynôme et qui retourne sa représentation sous forme d'un tableau.
3. Écrire une méthode `degrePolynome` qui retourne le degré d'un polynôme. Par convention, la méthode retournera -1 dans le cas du polynôme nul.
4. Écrire une méthode `creerMonome` qui fabrique la représentation d'un monôme aX^n .
5. Écrire une méthode `coefficient` qui prend en paramètres un polynôme et un entier positif (correspondant à un exposant), et qui retourne le coefficient correspondant. Pour un entier excédant la taille du tableau représentant le polynôme, la méthode doit retourner zéro.
6. Écrire une méthode d'affichage `affichePolynome`. Par exemple, elle affichera le polynôme $2X^2 - 3,5$ de la façon suivante : `2.0*X^2-3.5`.

Exercice 3: *Fonctions arithmétiques simples*

1. Écrire les méthodes de somme et de soustraction de polynômes.
2. Écrire la méthode de multiplication de deux polynômes. On rappelle que si $P = \sum_{i=0}^n a_i X^i$ et $Q = \sum_{j=0}^m b_j X^j$, alors $P \cdot Q = \sum_{i=0}^{n+m} (\sum_{k=0}^i a_k b_{i-k}) \cdot X^i$. La méthode pourra utiliser deux boucles *for* imbriquées.

Exercice 4: *Division euclidienne*

Les polynômes à coefficients réels sont munis d'une opération de division euclidienne analogue à celle des entiers. On a la propriété suivante : si A et B sont deux polynômes (avec $B \neq 0$), il existe un unique couple de polynômes Q (le quotient) et R (le reste) tels que :

- le degré de R est strictement inférieur à celui de B ,
- $A = BQ + R$.

Le quotient et le reste d'une division euclidienne peuvent se calculer à la main en posant la division à la manière d'une division entière. Voici un exemple

d'un tel calcul, la division de $2X^4 + 2X^2 + 1$ par $2X^2 + X$:

$$\begin{array}{r|l}
 \begin{array}{r}
 2X^4 \qquad \qquad + 2X^2 \qquad \qquad + 1 \\
 \underline{2X^4 + X^3} \\
 -X^3 \qquad + 2X^2 \qquad + 1 \\
 \underline{-X^3 - (1/2)X^2} \\
 (5/2)X^2 \qquad \qquad + 1 \\
 \underline{(5/2)X^2 + (5/4)X} \\
 -(5/4)X + 1
 \end{array} &
 \begin{array}{l}
 2X^2 + X \\
 \hline
 X^2 - (1/2)X + (5/4)
 \end{array}
 \end{array}$$

Programmer la methode de division euclidienne. Elle repose sur une boucle `while` pour laquelle on exhibera un invariant. Justifier la correction de la methode.

Indication : on pourra utiliser une variable contenant successivement chaque reste intermediaire de l'operation de division (dans l'exemple ci-dessus : $2X^4 + 2X^2 + 1$, $-X^3 + 2X^2 + 1$, $(5/2)X^2 + 1$ et enfin le reste resultat $-(5/4)X + 1$).

Remarque d'ordre pratique : comme dans la suite nous serons uniquement interesses par le reste de la division, la methode se contentera de retourner le reste.

Exercice 5: *Algorithme d'Euclide*

On s'interesse au pgcd d'un couple de polynomes $\neq (0, 0)$. Attention, dans le cas des polynomes, il n'y a pas unicite du pgcd. Il y a une infinite de pgcd possibles pour un couple de polynomes, qui different par un coefficient de proportionalite reel (non nul). On peut donc deduire l'ensemble des pgcd de deux polynomes a partir d'un pgcd quelconque.

Le but de l'exercice est de programmer l'algorithme d'Euclide (comme celui de l'exercice ??) pour les polynomes. La methode est essentiellement la meme que pour les entiers.

1. Quel argument permet de demontrer la terminaison de l'algorithme dans le cas des polynomes ?
2. Ecrire la methode *pgcd* basee sur cet algorithme (elle est construite autour d'une boucle `while`).
3. Exhiber un invariant pour la boucle `while` permettant de justifier la correction du programme.
4. Rediger un programme complet qui calcule un pgcd de deux polynomes. On choisit d'afficher le pgcd dont le coefficient dominant est egal a 1.