Université Paris XII
IUT de Fontainebleau
Département informatique
Algorithmique Programmation
Troisième semestre
2006–2007

PARTIEL 1

Seuls sont autorisés, à titre de documents, les listings comprenant explicitement le nom de l'étudiant (et celui-ci seulement) et les notes manuscrites.

Les deux exercices sont indépendants mais ils doivent être présentés dans l'ordre (prévoir deux pages par exercice).

Exercice 1.- (**Polynômes**)

Un polynôme à coefficients entiers, par exemple :

$$p(x) = 5.x^{10} + 9.x^7 - x - 10$$

est stocké comme liste chaînée de termes (5, 10), (9, 7), (-1, 1), (-10,0).

- 1^o) Écrire une classe Java `Term` qui contient un terme (de la liste chaînée).

[*On laissera de la place pour ajouter les méthodes qui se révéleront nécessaires dans la question suivante.*]

- 2^o) Écrire une classe `Polynom` dont l'attribut est un terme (de liste chaînée) et dont les méthodes sont :

1. le constructeur par défaut,
2. une méthode `addTerm(int c, int e)` qui ajoute le terme $c.x^e$,
3. * une méthode `removeTerm(int e)` qui supprime le terme d'exposant e s'il existe,
4. ** une méthode `add(Polynom)` qui ajoute un polynôme au polynôme (le résultat doit être sous forme réduite :

   $$(x^3 + 2.x^2) + (x^2 - 10.x + 1) = x^3 + 3.x^2 - 10.x + 1.$$

   bien entendu),
5. une méthode `print()` qui affiche le polynôme sous la forme :

   ```
   0x3 + 3x2 - 10x1 + 1x0.
   ```

- 3^o) Écrire un programme de test.

Exercice 2.- (**Dessinons des cercles**)

Écrire une application Java qui affiche un cadre (fermant) sur lequel lorsqu'on clique sur deux emplacements, le cercle de centre le premier endroit cliqué et passant par le deuxième endroit cliqué est dessiné.

[*Il faut donc garder la trace du fait que l'on a déjà cliqué sur le centre ou non.*]

1 Documentation

1.1 Ellipse

La classe :

`Ellipse2D.Double`

du paquetage :

`java.awt.geom`

a pour constructeur :

```
Ellipse2D.Double(Double x, Double y, Double hauteur,
Double largeur)
```

où x et y désignent les coordonnées du coin inférieur gauche du rectangle de hauteur `hauteur` et de largeur `largeur` dans lequel l'ellipse est inscrite.

Si `hauteur = largeur`, on a un cercle.

La méthode :

`draw(Geom)`

de la classe abstraite :

`Graphics2D`

(la seule façon d'en obtenir une instance est de convertir de façon explicite un objet de la classe abstraite `Graphics`) permet de dessiner une ellipse.

1.2 Événements souris

Les événements souris ont pour classe d'écoute :

```
public interface MouseListener
{
  void mousePressed(MouseEvent event);
    // Appelee lorsqu'on a appuye sur un bouton de souris
  void mouseReleased(MouseEvent event);
    // Appelee lorsqu'on a relache un bouton de souris
  void mouseClicked(MouseEvent event);
    // Appelee lorsqu'on a cliqu sur un bouton de souris
  void mouseEntered(MouseEvent event);
    // Appelee lorsque la souris penetre dans un composant
  void mouseExited(MouseEvent event);
    // Appelee lorsque la souris quitte le composant
}
```

dont **toutes** les méthodes doivent être surchargées.

La classe `MouseEvent` possède les deux méthodes :

```
int getX()
int getY()
```

qui renvoient l'abscisse et l'ordonnée (en pixels) du curseur de la souris au moment de l'événement.

La mise à l'écoute d'un événement souris sur un composant s'effectue grâce à la méthode :

`addMouseListener()`