TP de Programmation Orientée Objet N°2
ILOG1 *

15 décembre 2004

1 Pour les débutants en POO et/ou JAVA...

L'objectif de ce TP est d'acquérir une bonne maîtrise dans l'écriture complète (code source *et* commentaires) d'une classe en JAVA.

1.1 Travail à faire pendant la séance de TP

1. En vous inspirant du travail effectué pour la classe `PileElem`, définissez une nouvelle classe `PilePoint` dans laquelle les éléments de la pile sont des instances de la classe `Point` de la librairie standard JAVA (cf. documentation à l'adresse `http://java.sun.com/j2se/1.4.2/docs/api/java/awt/Point.html`).
2. Définissez une nouvelle classe `TestPilePoint`. Cette classe, qui contiendra une unique méthode de signature `public static void main(String[] args)`, sera utilisée pour produire un programme JAVA exécutable permettant de tester les fonctionnalités de la classe `PilePoint`.
3. Ecrivez la documentation de la classe `PilePoint`.

1.2 Travail à rendre

Les deux fichiers : `PilePoint.java` et `TestPilePoint.java`.

2 Pour ceux qui connaissent déjà le langage JAVA...

2.1 Travail à faire pendant la séance de TP

En vous inspirant de la classe `PureListeElem` que vous avez définie lors du dernier TP, définissez les nouvelles classes `PureListePoint` et `CellPoint` (inspirée de la classe `CellElem`) permettant de représenter des listes dont les éléments sont des instances de la classe `Point` de la librairie standard JAVA (cf. documentation à l'adresse `http://java.sun.com/j2se/1.4.2/docs/api/java/awt/Point.html`). De plus, votre implémentation devra respecter les contraintes suivantes :

- La classe `PureListePoint` devra implémenter les méthodes `equals` et `toString`, et, si nécessaire, les méthodes `clone` et `hashcode`.
- Les contraintes exprimées sous forme de commentaires dans le code ci-après doivent satisfaites :

```
PureListePoint lp1;
Point p1, p2;
int i, j;

lp1 = new PureListePoint();
... // code quelconque...
p1 = new Point(12, 5);
lp1 = lp1.cons(p1);
lp1 = lp1.cons(new Point(-1, 6));
```

*M. Champesme – Département d'Informatique – Institut Galilée

```
p2 = new Point(12, 5);
i = lp1.rechercher(p1);
j = lp1.rechercher(p2);
// à cet endroit on doit avoir i == 2 et j == 2
... // code arbitraire ne modifiant pas la référence
... // contenue dans lp1
p1.x = 0;
p1.y = 1;
j = lp1.rechercher(new Point(12, 5));
// à cet endroit on a encore j == 2
```

– Vos classes doivent être *entièrement* commentées au format “javadoc” : commentaire de classe, un commentaire pour chaque méthode (y compris commentaire pour chaque paramètre et pour chaque valeur de retour).

Définissez une nouvelle classe `TestListePoint` permettant de tester l’essentiel des fonctionnalités de la classe `PureListePoint`.

2.2 Travail à rendre

Les trois fichiers : `PureListePoint.java`, `CellPoint.java` et `TestListePoint.java`.

3 Note sur le travail à rendre

– Même lorsque le travail a été fait en groupe, chaque étudiant doit rendre sa propre version du travail demandé.
– Le travail demandé doit être rendu de préférence, par courrier électronique à l’adresse `mailto:Marc.Champesme@lipn.univ-paris13.fr` en précisant dans le sujet (ou objet) du message : “POO” (pour programmation orientée objet), “CR TP2” (pour compte rendu du TP n°2) et votre nom.