Travaux Dirigés de Java n°3

Cours de Java

—IR 2—

Mars 2003

Dans cette séance nous utiliserons au mieux (en prenant garde aux espaces mémoires et à la complexité des opérations) les classes de conteneurs de `java.util`. Nous utilisons également les classes de flots (`java.io`).

▸ **Exercice 1.** *Ecrire un programme qui prend une suite de mots en argument sur sa ligne de commande, qui compte le nombre de fois que chaque mot apparaît, et qui affiche la liste des mots avec le nombre d'occurrences de chacun.*

▸ **Exercice 2.** *Même exercice, mais on ajoute des options :*

- `-a` *: affiche la liste des mots et de leur nombre d'occurrences dans l'ordre lexicographique ;*
- `-b` *: affiche la liste des mots et de leur nombre d'occurrences dans l'ordre lexicographique inverse;*
- `-e` *: affiche la liste des mots et de leur nombre d'occurrences dans l'ordre militaire (les mots les plus courts d'abord, si deux mots ont la même longueur on applique l'ordre lexicographique) ;*
- `-f` *: affiche la liste des mots et de leur nombre d'occurrences dans l'ordre militaire inverse ;*
- `-x` *: dans l'ordre décroissant des nombres d'occurrences des mots ;*
- `-y` *: dans l'ordre croissant des nombres d'occurrences des mots ;*

▸ **Exercice 3.** *Soit la classe suivante, représentant des graphes orientés avec un nœud initial :*

```
public class Graph {
  private Node start;
  public void setStart(Node n) {
    start = n;
  }
```



```
  public class Node {
    private ArrayList transitions;
    public Node() {
       transitions = new ArrayList();
    }
    public void addTrans(Node dest) {
      transitions.add(dest);
    }
    public Iterator transitions() {
      return transitions.iterator();
    }
  }
}
```

Écrire une méthode `main` *dans laquelle vous créerez un graphe à cinq états, avec une circularité. Ajoutez à la classe* `Graphe` *une méthode permettant d'afficher le graphe. Voici un exemple de ce qu'on souhaite obtenir :*

```
Noeud initial : n1
n1: n2 n3
n2: n4
n3:
n4: n3 n1 n5
n5: n5
```

L'affichage indique le nœud initial, puis une ligne par nœud du graphe, chaque ligne contenant un identificateur pour le nœud suivi de deux points et des nœuds vers lesquels il possède un arc.

▸ **Exercice 4.** *Modifiez votre fonction d'affichage pour qu'elle prenne en argument un flot de caractères en écriture. Utilisez-la pour réaliser la sauvegarde d'un graphe dans un fichier texte. Écrivez l'opération inverse (la lecture d'un graphe à partir d'un fichier texte).*

▸ **Exercice 5.** *On souhaite maintenant sauvegarder les graphes dans des flots d'octets (fichiers binaires). Ajoutez une méthode* `save` *à la classe* `Graph` *pour faire cette sauvegarde. Écrivez aussi la méthode de lecture !*