TP n°1

Contexte applicatif

La série de Tps vise à réaliser pas à pas un logiciel de vente par correspondance. Les principales fonctionnalités de l'application sont :

- constitution d'un stock d'articles.
- enregistrement du stock sur disque
- constitution d'un panier client
- préparation de commandes

L'interface utilisateur sera réalisée à travers une interface graphique basique.

Objectifs

Cette série de Tps a pour objectif de visiter et expérimenter les principaux concepts du cours :

- conception de classes, création d'objets, écriture et documentation de méthodes, envoi de messages et transmission de paramètres
- utilisation de collections, de « design pattern »
- lecture et écriture d'objets sur fichier disque
- héritage, classes abstraites, interfaces et polymorphisme
- exceptions
- distribution de logiciel

La structure générale du projet est constituée :

- du package qui regroupe les classes qui gèrent l'interface utilisateur
- du package qui regroupe les classes métiers (coeur de l'application)
- du package qui regroupe les classes qui gèrent la persistance des objets

Enoncé

Les articles proposés par le magasin sont caractérisés par une référence (représentée par un chaîne de caractères formée de chiffres de longueur 8 au maximum), une dénomination, un prix hors taxe, un taux de TVA.

Question 1

- Créer le projet `tp1` (sous bluej ou netbeans)
- Créer la classe `Article` avec ses constructeurs, accesseurs et une définition de la méthode `toString()` qui retourne une chaîne de caractères décrivant l'article sous la forme :

  `"[ref : `*`uneref`*`, nom: `*`unNom`*`, prixHT: `*`unPrix`*`,TVA: `*`laTVA`*`]"`

Question 2

Créer une classe `Stock` qui rassemble l'ensemble des articles créés dans une même structure de données (Collection). Il faut aussi représenter le nombre d'exemplaires disponibles pour chaque article. On choisira dans le package `java.util` la structure la plus adaptée parmi :

`ArrayList, HashMap, HashSet.`

En plus de ses constructeur et accesseurs, cette classe devra comporter des méthodes :

pour ajouter un article au stock.
pour le supprimer du stock
pour retourner un article connaissant sa référence
pour retourner le nombre d'exemplaires associé à un article

Question 3

L'interface utilisateur permettant la saisie et l'affichage sera réalisée à travers une interface graphique simple (note 1). Les méthodes de saisie et affichage seront réparties dans des classes spécifiques.

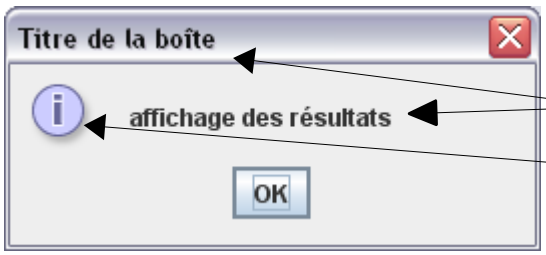
Question 4

Afin de créer un jeu de données persistant, on construit une classe, contenant une méthode main, dont le rôle est de créer un fichier dont chaque enregistrement est un objet, instance d'Article.

Ecrire une méthode `article2Fichier` qui saisit les caractéristiques d'un article via une interface graphique simplifiée (note 1) et qui les enregistre dans un fichier (note 2).

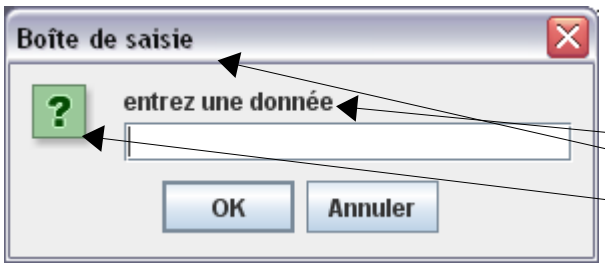
Note 1 : utilisation de la classe `java.swing.JOptionPane`

Pour présenter des résultats :



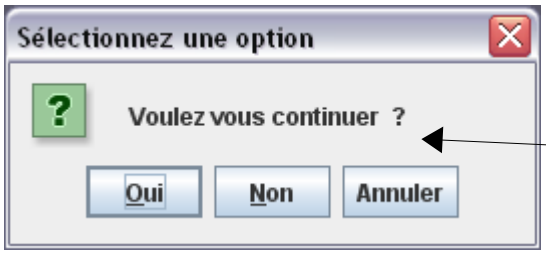

```
JOptionPane.showMessageDialog(null,
                              "affichage des résultats",
                              "Titre de la boîte",
                              JOptionPane.INFORMATION_MESSAGE
);
```

Pour saisir des valeurs :




```
String donnee =
JoptionPane.showInputDialog(null,
                            "entrez une donnée",
                            "Boîte de saisie",
                            JOptionPane.QUESTION_MESSAGE);
```

Pour terminer une suite de saisies à l'aide d'une boîte de confirmation :




```
int reponse =
  JOptionPane.showConfirmDialog(null,
                                " Voulez vous continuer  ?");
```

Note 2 :

La classe `ObjectOutputStream` possède la méthode `writeObject()` pour écrire une instance d'objet, dont la classe implémente l'interface `Serializable`, sur un fichier.

```
FileOutputStream fos = new FileOutputStream("monFichier.txt");
ObjectOutputStream out = new ObjectOutputStream(fos);
out.writeObject(article); // article, une instance de la classe Article
```

```
out.close();
```

pour lire le fichier :

```
FileInputStream fis = new FileInputStream("monFichier.txt");
ObjectInputStream in = new ObjectInputStream(fis);
article=(Article)in.readObject();
in.close();
```

Question 5

Ecrire la méthode `fichier2Stock` qui, à partir du fichier d'articles, génère un stock d'articles.

Question 5

Ecrire une classe `Client` **qui** crée un fichier d'articles puis constitue un stock de ces articles et enfin les affiche.