


Ecole Supérieure Privée d'Ingénierie et de Technologie	
Matière : Programmation orientée objet et langage Java	
Enseignants : Sofien Gharbi Saif Braham Rojdi Rekik Mehdi Hendili Bessem Hmidi Ibtihel Shimi	
Documents : Non autorisés Session : Principale	Date : 06/02/2009 Durée : 1h30

Examen

Nous souhaitons réaliser une solution de gestion d'une agence de location de Voitures. Les classes principales de ce programme sont : la classe **Voiture**, la classe **Client** et la classe **Agence**.

Une voiture est caractérisée par son *immatriculation*, sa *marque* et son *prix*. Deux voitures sont égales s'ils ont la même immatriculation et la même marque.

Q1. Compléter la classe **Voiture** ?

Un client est caractérisé par un *code*, un *nom* et un *prénom*. Deux clients sont égaux s'ils ont le même code.

Q2. Compléter la classe **Client** ?

Cette agence a mis à la disposition de ces clients la possibilité de louer plusieurs voitures, pour cela on a créé une classe **ListVoitures** qui permet de stocker un ensemble de voiture dans une liste. Cette classe permet d'ajouter, de supprimer une voiture et d'afficher la liste des voitures.

Q3. Compléter la classe **ListVoitures** ?

Cette agence offre à ces clients la possibilité de choisir les voitures à louer en fonction de différents critères. Il est possible de sélectionner dans la liste des voitures à louer toutes les voitures satisfaisant un critère donné. On définit l'interface Critère ainsi :

```
public interface Critere {

    boolean estSatisfaitPar(Voiture v);

}
```

Q4. Compléter la classe **CritereMarque** (Choix selon la marque) ?

Q5. Compléter la classe **CriterePrix** (Choix selon le prix) ?

Une agence est caractérisée par un *nom*, un parking de type ListVoiture qui permet d'enregistrer les voitures de l'agence de location et un ClientVoitureLoue de type Map (la partie clé est Client, la partie valeur est ListVoitures) qui permet d'enregistrer pour chaque client la liste des voitures actuellement louées.

Les fonctions standard pour la gestion de location de voiture sont :

- louer une voiture à un client.
- retourner une voiture louée par un client,
- retourner la liste des voitures selon un critère défini
- retourner l'ensemble des clients qui ont loué une (des) voiture(s)
- retourner la liste des voitures actuellement en état de location
- afficher les clients et leurs voitures louées.

Q6. Compléter la classe **Agence** ?

Q7. Compléter la méthode qui effectue le tri selon le code du client ?

Q8. Compléter la méthode qui effectue le tri selon le nom du client ?

Remarque :

- Vous avez, en annexe, **quelques** codes incomplets pouvant vous guider à implémenter les classes. S'il y a d'autres classes à développer n'hésitez pas à les écrire.

```

public class Voiture {
    private int immatriculation;
    private String marque;
    private float prixLocation;
    public Voiture(int immatriculation, String marque,
float prixLocation) {
        // à compléter
    }
    public int getImmatriculation() {
        return immatriculation;
    }
    public void setImmatriculation(int immatriculation) {
        this.immatriculation = immatriculation;
    }
    public String getMarque() {
        return marque;
    }
    public void setMarque(String marque) {
        this.marque = marque;
    }
    public float getPrixLocation() {
        return prixLocation;
    }
    public void setPrixLocation(float prixLocation) {
        this.prixLocation = prixLocation;
    }
    public int hashCode() {
        //à completer
    }
    public boolean equals(Object obj) {
        //à completer
    }
    public String toString(){
        //à completer
    }
}

```

```

public class Client {
    private int code;
    private String nom;
    private String prenom;
    public Client(int code, String nom, String
prenom) {
        //à compléter
    }

    public int getCode() {
        return code;
    }
    public void setCode(int code) {
        this.code = code;
    }
    public String getNom() {
        return nom;
    }
    public void setNom(String nom) {
        this.nom = nom;
    }
    public String getPrenom() {
        return prenom;
    }
    public void setPrenom(String prenom) {
        this.prenom = prenom;
    }
    public int hashCode() {
        //à completer
    }
    public boolean equals(Object obj) {
        //à completer
    }
    public String toString(){
        //à completer
    }
}

```

```

public class CritereMarque implements Critere {
    private String marque;
    public CritereMarque(String marque) {
        this.marque = marque;
    }
    public boolean estSatisfaitPar(Voiture v) {
        // à compléter
    }
}

```

```

public class CriterePrix implements Critere {
    private float prix;
    public CriterePrix(float prix) {
        this.prix = prix;
    }
    public boolean estSatisfaitPar(Voiture v) {
        //à Compléter
    }
}

```

```

public class ListVoitures {
    private List<Voiture> voitures;
    public ListVoitures(List<Voiture> voitures) {
        //à compléter
    }
    public ListVoitures() {
        //à compléter
    }
    public List<Voiture> getVoitures() {
        //à compléter
    }
    public void setVoitures(List<Voiture> voitures) {
        //à compléter
    }

    public void ajoutVoiture(Voiture v) throws
    VoitureException{

        //à compléter
    }
    public void supprimerVoiture(Voiture v) throws
    VoitureException{
        // à compléter
    }
    public Iterator<Voiture> iterateur(){
        return voitures.iterator();
    }
    public int size(){
        return voitures.size();
    }
    public void affiche(){
        // à completer
    }
}

```

```

public class Agence {
    private String nom;
    private ListVoitures vs;
    private Map<Client, ListVoitures> ClientVoitureLoue;
    public Agence(String nom) {
        //à completer
    }
    public void ajoutVoiture(Voiture v) throws
    VoitureException{
        // à completer
    }
    public void supVoiture(Voiture v) throws
    VoitureException{
        //à completer
    }

    public void loueClientVoiture(Client cl, Voiture
    v) throws VoitureException{
        // à completer
    }
    public void retourClientVoiture(Client cl , Voiture v)
    throws VoitureException{
        // à completer
    }

    public List<Voiture> selectVoitureSelonCritere(Critere
    c){
        //à completer
    }
    public Set<Client> ensembleClientsLoueurs(){
        // à completer
    }
    public Collection<ListVoitures>
    collectionVoituresLouees(){
        // à completer
    }
    public void afficheLesClientsEtLeursListesVoitures(){
        //à compléter
    }
    public Map<Client, ListVoitures> triCodeCroissant(){
        // à completer
    }
    public Map<Client, ListVoitures> triNomCroissant(){
        // à completer
    }
}

```