TP 2 partie 2 (POO)

Création de classes

Le but de ce TP est de définir et de manipuler quelques classes élémentaires. Il se compose de deux exercices, le premier a pour but de définir une classe proposant des méthodes de classe pour les saisies. Le deuxième consiste à définir des classes pour une application.

Une classe est définie par le mot clé `class` :

```
class <nom de la classe> {
  <définitions de variables d'instance>     // utilisez le mot clé private pour forcer l'encapsulation
  <définitions de variables de classe>      // utilisez les mot clés static et private
  <définitions des méthodes d'instance>  // vous pouvez utilisez public
  <définitions de méthodes de classe>       // utilisez le mot clé static
}
```

Une méthode est définie de la manière suivante :

```
static | public <type retourné> <nom méthode> ( <liste d'arguments>) {
            <déclarations et instructions>
}
```

Le mot clé `return` délivre la valeur de la méthode. Si la méthode ne délivre rien (pas d'utilisation du mot clé `return`) le type retourné par la méthode est `void` (ce qui signifie vide). L'annexe donne un exemple de définition de classe.

Exercice /

Cet exercice concerne la gestion (très simplifiée) d'agences de location d'automobiles. Chaque agence gère un groupe de clients, un groupe de voitures et les locations.

Question 1/ Créez la classe `Voiture`. Une voiture se caractérise par son immatriculation (`String`), son modèle (`String`), son nombre de kilomètres parcourus (`int`) et son tarif de location au kilomètre (`float`). Ecrivez un constructeur qui demande la saisie au clavier des caractéristiques de la voiture créée. Ecrivez des méthodes d'instance pour accéder aux variables d'instance et une méthode d'instance pour l'affichage.

Testez cette classe et ses méthodes.

Question 2/ Une instance d'`ArrayList` est un tableau dynamique. Dans un main, déclarez et utilisez une instance d'`ArrayList` d'au plus 10 voitures (voir la classe `ArrayList` dans la documentation JAVA en ligne : http://docapi.iutlan.univ-rennes1.fr/). Insérez quelques voitures, affichez l'`ArrayList`, supprimez quelques voitures et affichez l'`ArrayList` etc.

Question 3/ Créez la classe `Client`. Un client se caractérise par un numéro (`int` généré automatiquement lors de chaque création), d'un nom (`String`) et d'un domicile (`String`). Réfléchissez au moyen de générer automatiquement le numéro des clients.

Testez cette classe par la création de clients et de leur affichage.

Question 4/ Proposez une définition pour la classe `Date`. Une date se caractérise par un jour (`int`), un mois (`int`) une année (`int`). Le constructeur demande les informations au clavier, et on souhaite pouvoir afficher une instance de `Date`. Testez cette classe.

Question 5/ Proposez une définition pour la classe `Location`. Une location se caractérise par un numéro (`int`) généré automatiquement, une voiture (`Voiture`), un client (`Client`), une date de location (`Date`) et une date de retour (`Date`) et un nombre de kilomètre. Le constructeur admet la voiture, le client et la date de location comme paramètres. La date de retour est mis à la valeur `null` et le nombre de kilomètres parcourus (inconnu à la création) est en fait le kilométrage initial de la voiture louée. Une méthode pour l'affichage doit être prévue (bien distinguer le cas où la date de retour est `null` ou non, car dans le premier cas il s'agit d'une location en cours , et d'une location terminée dans le deuxième cas). Testez cette classe.

Question 6/ Proposez une définition pour la classe `Agence`. Les voitures, les clients et les locations de voitures sont conservés dans trois instances d'`ArrayList` distincts (de taille maximale 100). Le numéro d'un client et d'une location servira d'indice dans l'instance d'`ArrayList`. Proposez un constructeur qui initialise l'agence avec quelques voitures, clients et locations en cours. Ce constructeur ne devra pas faire de lecture au clavier, vous serez donc obligé de définir de nouveaux constructeurs pour les classes `Voiture`, `Date` et `Client`. Des méthodes pour l'affichage des voitures, clients et locations doivent être prévues .

Question 7/ Proposez et testez une méthode pour enregistrer une location (on doit demander le numéro du client, vérifier s'il est bien enregistré, lui proposer les véhicules disponibles et enregistrer sa location ou abandonner si le client n'est pas satisfait). Attention, définir cette méthode peut demander de rajouter d'autres méthodes.

Question 8/ Proposez et testez une méthode pour enregistrer le retour d'une location (on doit demander le numéro du client et l'immatriculation du véhicule, vérifier qu'il s'agit bien d'une location en cours, demander le nombre de kilomètres effectués et la date du retour, enregistrer ces renseignements puis afficher le prix à payer par le client à l'écran). Attention, définir cette méthode peut demander de rajouter d'autres méthodes.

Annexe

Voici une classe définissant des personnes :

```
class Personne
{
	// variable d'instance
	// donc privees

	private String nom;
	private int age;

	// variable de classe
	// donc static

	static private int nb=0;

	// constructeur : de meme nom que la classe
	// eventuellement plusieurs constructeurs

	public Personne(String lenom, int lage){
		nom=lenom;
		age=lage;
		 nb++;
	}

	public Personne(){
		nom="essai";
		age=10;
		nb++;
	}
	// Methodes d'instance

	public int obtenirAge(){
		return this.age;
	}

	public String obtenirNom(){
		return this.nom;
	}

	public void modifierNom(String nouveauNom){
		this.nom=nouveauNom;
	}

	public void modifierAge(int nouvelAge){
		this.age = nouvelAge;
	}

	// Methode de classe

	static int obtenirNb(){
		return nb;
	}

}
```