POO : TD agence

rental::Vehicle
- brand : String - model : String - productionYear : int - dailyrentalPrice : double
+Vehicle(brand : String, model : String, productionYear : int, dailyRentalPrice : double) + getBrand() : String + getModel() : String + getProductionYear() : int + getDailyRentalPrice() : double + equals(o : Object) : boolean + toString() : String

La méthode `toString()` fournit une chaîne de caractères qui représente le descriptif du véhicule qui reprend les valeurs des différents attributs.

Agence de location

La classe `rental.RentalAgency` est définie ainsi :

rental::RentalAgency
- allVehicles : List<Vehicle>
+ RentalAgency() + addVehicle(c : Vehicle) + getAllVehicles() : List<Vehicle> + displayAllVehicles() + removeVehicle(c : Vehicle) ...

Q 4 . Donnez le code de la méthode `removeVehicle()` de la classe `RentalAgency` qui permet de supprimer un véhicule à l'agence. Cette méthode doit déclencher une exception `UnknownVehicleException` si le véhicule n'est pas géré par l'agence.

On suppose la classe d'exception `UnknownVehicleException` définie dans le paquetage `rental`.

Q 6 . Donnez le code d'une méthode `selectByMaxPrice100()` de la classe `RentalAgency` dont le résultat est la liste de tous les véhicules gérés par l'agence dont le prix de location à la journée est inférieur à 100€.

```
public List<Vehicle> selectByMaxPrice100() {
    List<Vehicle> Res = new ArrayList<Vehicle>();
    for (Vehicle v : this.getAllVehicles) {
        if (v.dailyRentalPrice < 100) {
            Res.add(v)
        }
}

Return Res;
```

Q 10 . Appliquez votre solution pour les trois filtres évoqués ci-dessus, que faut-il faire ? Que deviennent les méthodes `selectByMaxPrice100()` et `selectByBrandTimoleon()` ?

```
public interface Filter {
    public boolean select(Vehicle v);
}

public class PriceFilter implements Filter {
    private int price = 100;
    public void setPrice(int p) {
        this.price = price
    }
}

public boolean     select(Vehicle v) {
    return this.price > v.dailyRentalPrice();
}
```

Q 14 . En supposant que la référence agency est de type `RentalAgency` et a été initialisée, donnez la ou les lignes de code permettant d'afficher tous les véhicules de cette agence dont le prix est inférieur à 85€.

```
PriceFilter pf = new PriceFilter();
pf.setPrice(85);
agency.select(pf);
```

Q 15 . On peut naturellement souhaiter faire des « intersections de filtres », ce qui revient à appliquer le **et** logique entre des filtres. Par exemple sélectionner les véhicules produits après 2020 et dont le prix de location est au maximum 100€.

La solution précédente permet-elle de répondre à cette situation ?
Si oui, comment (et appliquez votre proposition à l'exemple donné) ?
Si non, pourquoi ?

Oui, on peut créer un filtre à deux attributs