UNIVERSITÉ LE HAVRE

UFR
Sciences et
Techniques

Licence 2

Java

21 mai 2021
Durée : 2 heures
Documents, téléphones et calculatrices non autorisés.
Il est possible d'utiliser les réponses à une question non traitée pour résoudre les autres questions.

I Une agence de location de voitures offre à ses clients la possibilité de choisir la voiture louée en fonction de différents critères. Les voitures sont définies par une marque, un nom de modèle, une année de production et un prix de location à la journée. Pour simplifier les deux premiers paramètres sont des objets de la classe `String`, le troisième est de type `int` et le dernier de type `double`. Deux voitures sont considérées égales si tous leurs attributs sont égaux.

I.1 On considère la classe suivante :

```
public class Voiture {
  private String marque;
  private String modele;
  private int anneeProd;
  private double prix;

  public Voiture(String brand, String model, int productionYear, int price) {...}
  public String marque() {...}
  public String modele() {...}
  public int anneeProd() {...}
  public double prix() {...}
  public String toString() {...}

  public boolean equals(Object o) {
  // ToDo
  }
}
```

Définir la méthode `equals()`.

I.2 Soit maintenant la classe

```
public class AgenceLoc {
    private String nomAgence;
    private List<Voiture> parcAuto;

    public AgenceLoc(String n) {
    // ToDo
    }
```

Écrire le constructeur `AgenceLoc()` à un paramètre.

I.3 Écrire une méthode `add()` de la classe `AgenceLoc`, qui permet d'ajouter une voiture à l'agence.

I.4 Écrire une méthode `supprime()` de la classe `AgenceLoc`, qui permet de supprimer une voiture à l'agence.

I.5 On souhaite pouvoir sélectionner parmi les voitures à louer toutes les voitures satisfaisant un critère donné. Écrire une méthode `selectionParMarque()` dans la classe `AgenceLoc` dont le résultat est la liste des véhicules de l'agence dont la marque est la même que celle passée en paramètre de la méthode. Vous devez utiliser une boucle "`foreach`".

I.6 Écrire une méthode `selectionParPrix()` dans la classe `AgenceLoc` dont le résultat est la liste des véhicules de l'agence dont le prix de location

25 Rue Philippe Lebon

B.P. 540

76058 Le Havre Cedex

est inférieur à celui passé en paramètre de la méthode. Vous devez utiliser une boucle "foreach".

I.7 Comparez ces deux méthodes, qu'en pensez-vous ? Imaginez d'autres critères pouvant être sélectionnés.

I.8 Predicate<T> est une interface fonctionnelle qui représente un prédicat qui attend en paramètre un argument de type T et renvoie un booléen. Écrire une méthode filtrerSurCritere() dans la classe AgenceLoc dont le résultat est une liste de véhicules respectant le prédicat fourni en entrée. Vous devez utiliser des streams.

I.9 Écrire une méthode filtrerParMarque() dans la classe AgenceLoc dont le résultat est la liste des véhicules de l'agence dont la marque est la même que celle passée en paramètre de la méthode. Vous devez utiliser la méthode filtrerSurCritere() et le prédicat doit être écrit sous la forme d'une lambda.

I.10 Écrire une méthode filtrerParPrix() dans la classe AgenceLoc dont le résultat est la liste des véhicules de l'agence dont le prix de location est inférieur à celui passé en paramètre de la méthode. Vous devez utiliser la méthode filtrerSurCritere() et le prédicat doit être écrit sous la forme d'une référence de méthode.

II Donner un exemple complet de traitement d'exception que vous commenterez.

Annexe

L'interface List<E> est définie dans le package java.util et est implémentée par la classe java.util.ArrayList<E>. L'interface java.util.List<E> définit un certain nombre de méthodes :

- boolean add(E e) : Appends the specified element to the end of this list.
- boolean contains(Object o) : Returns true if this list contains the specified element.
- E get(int index) : Returns the element at the specified position in this list.
- boolean isEmpty() : Returns true if this list contains no elements.
- boolean remove(Object o) : Removes the first occurrence of the specified element from this list, if it is present.
- int size() : Returns the number of elements in this list.

public final class Collectors : Implementations of Collector that implement various useful reduction operations, such as accumulating elements into collections,

- static <T> Collector<T,?,List<T>>toList() : Returns a Collector that accumulates the input elements into a new List.