

Examen Java
Juin 2016
Professeur Abdessamad Belangour
SMI S6

Nous souhaitons faire une petite application console qui gère les trains, leurs wagons et leurs conducteurs. Un train se compose d'un certain nombre de Wagons qui peuvent être des de passagers, de Minerais (phosphates,) ou de marchandises. Ainsi, un train peut être :

- Un train de Passagers s'il est composé de wagons passagers seulement,
- Un train de Minerais s'il est composé de wagons Minerais seulement,
- Un train de Marchandise s'il est composé de wagons Marchandise seulement,
- Un train de Passagers/Marchandise s'il est composé de wagons passagers et de wagons Marchandise,

Vous devez fournir uniquement les parties de code demandées sur votre feuille :

- Sans réécrire toutes les classes mais juste les méthodes ou constructeurs demandées !!
- En respectant les signatures des méthodes et des constructeurs fournis !!

```
public abstract class Wagon {  
    private String code;  
    private String description;  
    // on suppose que les constructeurs, les getters et setters et toString() sont fournis  
}
```

```
public class WagonMarchandise extends Wagon{  
    private final static float volumeMax=72; // en mètres cube;  
    private float volume;  
    private static int nombreWagonsMarchandise=0;  
  
    public WagonMarchandise(String code, String description, float volume) { // à fournir }  
  
    // on suppose que le getter et setter du volume est fourni  
    // getters et ou setters pour VolumeMax et nombreWagonMarchandise à fournir  
    // toString à fournir  
}
```

```
public class WagonMinerais extends Wagon{  
    private final static float tonnageMax=22;// en tonnes  
    private float tonnage;  
    private static int nombreWagonsMinerais=0;  
  
    // on suppose que les constructeurs et les getters et setters et toString sont fournis
```

<pre> }</pre>
<pre> public class WagonPassagers extends Wagon{ private final static int nombrePassagersMax=50; private int nombrePassagers; private static int nombreWagonsPassagers=0; // on suppose que les constructeurs et les getters et setters et toString sont fournis }</pre>
<pre> public enum TypeTrain { Marchandise, Passagers, PassagersMarchandise, Minerai }</pre>
<pre> public class TrainWagonIncompatiblesException extends Exception{ public TrainWagonIncompatiblesException(String message) { super(message); } }</pre>
<pre> public class Personnel { private String matricule; private String nom; private String prenom; // on suppose que les constructeurs et les getters et setters et toString sont fournis }</pre>
<pre> public class Train { private String code; private Personnel conducteur; private TypeTrain type; private int longueur; private ArrayList<Wagon> wagons; public Train(String code, Personnel conducteur, TypeTrain type) { // à fournir } public Train(String code, Personnel conducteur, TypeTrain type, ArrayList<Wagon> wagons) { // à fournir } // getter et ou setter pour longueur à fournir // on suppose que les autres getters et setters sont fournis public void ajouterWagonPassagers(String code, String description, int nombrePassagers) throws TrainWagonIncompatiblesException { // cette méthode est fournie } public void ajouterWagonMinerai(String code, String description, float tonnage) throws TrainWagonIncompatiblesException { // à fournir } public void ajouterWagonMarchandise(String code, String description, float volume) throws TrainWagonIncompatiblesException { // on suppose que le code de cette méthode est fourni } public void ajouterWagon(Wagon wagon) throws TrainWagonIncompatiblesException { // à fournir } public boolean supprimerWagon(String code) { // à fournir } // on suppose que toString est fourni // la méthode afficherCharge() qui affiche la quantité, le nombre ou le volume de ce que transporte le train selon son type.</pre>

```
public void AfficherCharge() {    // à fournir }
```

Rappel des Questions :

- i. Dans la classe **WagonMarchandise** :
 0. Donner le code du **constructeur**
 1. Donner le code du getter et ou setter pour l'attribut *VolumeMax*
 2. Donner le code du getter et ou setter pour l'attribut *nombreWagonMarchandise*
 3. Donner le code de la méthode *toString*
- ii. Dans la classe **Train** :
 4. Donner le code du **premier constructeur**
 5. Donner le code du *deuxième constructeur*
 6. Donner le getter et ou setter pour l'attribut *longueur*
 7. Donner le code de la méthode *ajouterWagonMinerai* (Attention : l'ajout d'un wagon d'un type dans un train qui n'est pas du même type déclenche une exception)
 8. Donner le code de la méthode *ajouterWagon* (même remarque)
 9. Donner le code de la méthode *supprimerWagon*
 10. Donner le code de la méthode *afficherCharge* de la classe Train qui affiche la quantité, le nombre ou le volume de ce que transporte le train selon son type.