Examen écrit - Programmation par objets 1 (Java)

Tous documents autorisés

Dans ce sujet, nous étudions diverses classes liées à la gestion d'une exploitation forestière : des arbres, des essences forestières et des parcelles. Le code présenté à la figure 1 présente la classe `Arbre` et deux de ses sous-classes, `Chataignier` et `Erable`.

Question 1 *Pour les méthodes suivantes, indiquez si les déclarations des méthodes sont possibles (si la compilation se fera sans erreur) et s'il s'agit d'une surcharge (statique) ou d'une redéfinition (dynamique) :*
1. Méthodes `traite` *2. Méthodes* `taille` *3. Méthodes* `clone` *4. Méthodes* `greffe`

Question 2 *Si on suppose que les erreurs éventuellement découvertes dans la question précédente ont été corrigées, qu'affiche la méthode* `main` *de la classe* `Arbre` *?*

Question 3 *On vous donne la classe* `EssenceForestiere` *de la figure 2. Une essence forestière est une espèce d'arbre comme le châtaignier, l'érable ou le micocoulier.*

- *Complétez la classe* `Arbre` *par un attribut (d'instance) constant permettant de connaître l'essence d'un arbre et un constructeur avec un paramètre permettant de l'initialiser. Supposez que les accesseurs existent, et ne les écrivez pas.*
- *Complétez la classe* `Chataignier` *par un attribut de classe (valeur partagée par toutes les instances) constant initialisé par une instance de la classe* `EssenceForestiere` *représentant l'essence des châtaigniers. Cet attribut doit donner sa valeur à l'attribut d'instance de la classe* `Arbre` *(écrivez le constructeur approprié).*

On supposera pour la suite que la classe `Erable` *a été complétée de la même manière que la classe* `Chataignier`*.*

Question 4 *Une futaie est une parcelle forestière sur laquelle on ne cultive qu'une seule essence d'arbre. Ecrivez une classe* `Futaie_v1` *comprenant :*

1. *Des attributs représentant l'essence cultivée sur la parcelle, une localisation (chaîne de caractères) et une surface en hectares.*
2. *Un attribut permettant de stocker les arbres contenus dans cette futaie (pensez à l'initialiser quelque part).*
3. *Un constructeur avec un paramètre qui est l'essence cultivée dans la futaie.*
4. *Une méthode* `plante` *qui permet d'ajouter un arbre dans la futaie. L'arbre ajouté doit être de l'essence cultivée dans cette futaie. Risquez-vous d'ajouter un arbre d'une autre essence que celle du type d'arbre passé en paramètre ? Si c'est le cas, une exception (dont vous définirez la classe) doit être signalée.*
5. *Une méthode* `toString` *qui retourne une chaîne de caractères contenant la localisation, la surface et le nom commun de l'essence cultivée si elle peut être connue (sinon* `null`*).*

Question 5 *Ecrivez une classe* ***générique*** `Futaie_v2` *paramétrée par un type d'arbre comprenant :*

1. *Des attributs représentant une localisation (chaîne de caractères) et une surface en hectares.*
2. *Un attribut permettant de stocker les arbres contenus dans cette futaie (pensez à l'initialiser quelque part).*
3. *Une méthode* `plante` *qui permet d'ajouter un arbre dans la futaie. L'arbre ajouté doit être de l'essence cultivée dans cette futaie. Risquez-vous d'ajouter un arbre d'une autre essence que celle du type d'arbre passé en paramètre ? Si c'est le cas, une exception (dont vous définirez la classe) doit être signalée.*

4. *Une méthode* `toString` *qui retourne une chaîne de caractères contenant la localisation, la surface et le nom commun de l'essence cultivée si elle peut être connue (sinon* `null`*).*

Question 6 *Ecrivez une classe programme comprenant des instructions pour :*

1. *La création d'un châtaignier et d'un érable.*
2. *Deux futaies de châtaignier, respectivement avec chacune des versions (non générique, générique).*
3. *La plantation du châtaignier et de l'érable dans chacune des deux futaies.*
4. *L'affichage de chacune des deux futaies.*

Puis indiquez quelles instructions vont échouer et comment va se manifester l'erreur (notamment à quel stade, compilation ou exécution).

Question 7 *Ecrivez une classe* ***générique*** `Camion` *paramétrée par un type d'arbre comprenant :*

1. *Un attribut représentant l'immatriculation et le nombre maximum d'arbres que l'on peut charger.*
2. *Un attribut permettant de stocker les arbres chargés dans ce camion (pensez à l'initialiser).*
3. *Une méthode* `charge` *qui permet d'ajouter un arbre dans le camion.*

Les constructeurs et accesseurs sont supposés exister.

Question 8 *Ecrire une méthode qui enlève un arbre d'une futaie (de la classe générique) et le retourne (ou retourne* `null` *si la futaie est vide). Ajouter à la classe générique* `Camion` *une méthode* `emporte` *qui prend comme paramètre une futaie et charge sur le camion un des arbres de la futaie (en vérifiant que le nombre maximum n'a pas été atteint). Si* `Chataignier_Crenele` *est une sous-classe de* `Chataignier`*, un châtaignier crénelé doit pouvoir être emporté dans un camion prévu pour les châtaigniers.*

Question 9 *Ajouter à la classe générique* `Camion` *une méthode* `transborde` *qui prend comme paramètre un second camion et charge sur le second camion un des arbres du premier (en vérifiant que le nombre maximum n'a pas été atteint). Si* `Chataignier_Crenele` *est une sous-classe de* `Chataignier`*, on doit pouvoir transborder des châtaigniers crénelés dans un camion prévu pour les châtaigniers.*

Dans les deux questions suivantes traitant d'introspection vous utiliserez notamment les méthodes ci-dessous, qui peuvent signaler des exceptions que vous gèrerez :

— forName → ClassNotFoundException
— getConstructors → SecurityException
— newInstance → InstantiationException, IllegalAccessException, IllegalArgumentException, InvocationTargetException

Question 10 *Grâce à l'introspection (vous disposez dans les diapositives du cours de tous les éléments nécessaires), écrivez une méthode prenant comme paramètre une chaîne de caractères qui est un nom de classe et qui affiche les noms et listes de noms des types de paramètres de chacun des constructeurs de cette classe.*

Question 11 *Dans le cas spécifique où on ne passerait que des chaînes de caractères comme arguments aux constructeurs, complétez la méthode précédente avec les opérations suivantes :*

1. *saisir au clavier le numéro d'ordre d'un constructeur (dans l'affichage précédent).*
2. *demander à l'utilisateur des valeurs pour les paramètres de ce constructeur.*
3. *invoquer le constructeur.*

Indication : la méthode `newInstance` *de la classe* `Constructor` *prend un nombre variable de paramètres. On peut également l'utiliser en passant comme paramètre un tableau d'objets qui contient les différents paramètres réels.*

```
public class Arbre implements Cloneable{
   public void taille() throws MauvaiseSaisonTailleException {System.out.println("taille Arbre");}
   public Arbre clone() throws CloneNotSupportedException
   {System.out.println("clone Arbre");  return (Arbre)super.clone();}
   public void greffe(Arbre greffon) {System.out.println("greffe Arbre");}

   public static void main(String[] args)
         throws MauvaiseSaisonTailleException,CloneNotSupportedException {
      Arbre c1 = new Chataignier();
      Chataignier c2 = new Chataignier();
      Chataignier c3 = new Chataignier();
      Arbre e1 = new Erable();
      Erable e2 = new Erable();
      Object o1 = c1.clone();
      Object o2 = c2.clone();
      e1.taille();
      c1.greffe(c2);
      c2.greffe(c1);
      c2.greffe(c3);
      e2.traite(new Pesticide());
   }
}

public class Chataignier extends Arbre {
   public void taille() throws ArbreException {System.out.println("taille Chataignier");}
   public Chataignier clone() throws CloneNotSupportedException
   {System.out.println("clone Chataignier");  return (Chataignier)super.clone();}
   public void greffe(Chataignier greffon) {System.out.println("greffe Chataignier");}
}

public class Erable extends Arbre {
   public void taille() {System.out.println("taille Erable");}
   public Arbre clone() throws CloneNotSupportedException
      {System.out.println("clone Erable"); return (Erable)super.clone();}
   public void traite(Pesticide produit) {System.out.println("traite Pesticide");}
   public void traite(Engrais produit) {System.out.println("traite Engrais");}
   public boolean traite(Engrais produit) {System.out.println("traite Engrais boolean");}
}

public class ArbreException extends Exception{.....}
public class MauvaiseSaisonTailleException extends ArbreException {.....}

public class Pesticide {.....}
public class Engrais {.....}
```

FIGURE 1 – Classes `Arbre`, `Chataignier` et `Erable`

```
public class EssenceForestiere {
   private String nomCommun;
   private double volumeParDensite;
   public EssenceForestiere() {}
   public EssenceForestiere(String nomCommun, double volumeParDensite) {
      this.nomCommun = nomCommun;
      this.volumeParDensite = volumeParDensite;
   }
   public String getNomCommun() {return nomCommun;}
   public void setNomCommun(String nomCommun) {this.nomCommun = nomCommun;}
   public double getVolumeParDensite() {return volumeParDensite;}
   public void setVolumeParDensite(double volumeParDensite) {this.volumeParDensite = volumeParDensite;}
}
```

FIGURE 2 – Classe `EssenceForestiere`