Conception orientée objet

Application du cours 7 : Le musée

Cet énoncé vient en appui des diapositives du Cours.

I. Classe et interface déjà implémentées

```
public interface GestionTrophee {
        String tousLesTrophees();
        String lesTrophees(Gaulois proprietaire);
        void ajouterTrophee(Gaulois proprietaire, Equipement trophee);
        String donnerGauloisDonnateur();
        String donnerTrophees();
        String donnerInventaire();
}

public class Musee {
   private String nom;
   private int tarif;
   private GestionTrophee gestionnaireTrophee;

   public Musee(String nom, GestionTrophee gestionnaireTrophee) {
      this.nom = nom;
      this.gestionnaireTrophee = gestionnaireTrophee;
   }
   public String getNom() {
      return nom;
   }

   public int getTarif() {
      return tarif;
   }

   public void setTarif(int tarif) {
      this.tarif = tarif;
   }

   public void ajouterTrophee(Gaulois proprietaire, Equipement trophee) {
      gestionnaireTrophee.ajouterTrophee(proprietaire, trophee);
   }
   public String tousLesTrophees() {
      return gestionnaireTrophee.tousLesTrophees();
   }
   public String lesTrophees(Gaulois proprietaire) {
      return gestionnaireTrophee.lesTrophees(proprietaire);
   }
   public String donnerGauloisDonnateur() {
      return gestionnaireTrophee.donnerGauloisDonnateur();
   }
   public String donnerTrophees() {
      return gestionnaireTrophee.donnerTrophees();
   }
```

```
    public String donnerInventaire(){
        return gestionnaireTrophee.donnerInventaire();
    }
}
```

II. Travail à effectuer

1. Dans la classe « KeskonrixGestion », écrire l'attribut trophees comme une association entre un objet de type « Gaulois » (la clé) et une liste d'objets de la classe « Equipement ».

2. Compléter les méthodes :
   - *ajouterTrophee* qui place le nouvel équipement dans la liste de trophée d'un gaulois. Si le gaulois n'est pas dans les clés de la map, alors l'ajouter.
   - *tousLesTrophees* qui retourne une chaine contenant l'ensemble des trophées du musée classé selon leur donateur. Exemple de chaine : `System.out.println(musee.tousLesTrophees());`

```
Tous les trophées du musée sont :
Les trophées de Ordralfabétix sont :
- un casque

Les trophées de Astérix sont :
- un bouclier
- un casque
- une épée

Les trophées de Obélix sont :
- une épée
```

   - *lesTrophees* qui retourne une chaine contenant l'ensemble des trophées du musée d'un donateur en particulier. Exemple de chaine : `System.out.println(musee.lesTrophees(asterix));`

```
Les trophées de Astérix sont :
- un bouclier
- un casque
- une épée
```

3. Pour la compréhension sur les vues d'une map, on ajoute 2 attributs :
   - gauloisDonateur initialisé avec une vue sur les clés de l'association trophees,
   - collectionTrophees initialisé avec une vue sur les valeurs de l'association trophees.

   Créer les méthodes :
   - *donnerGauloisDonateur* qui renvoie une chaine contenant l'ensemble des gaulois ayant fait un don au musée. Exemple de

chaine :
`System.out.println(musee.donnerGauloisDonateur());`

```
Les gaulois ayant donné au moins un trophée au musée sont :
- Ordralfabétix
- Astérix
- Obélix
```

- *donnerTrophees* qui renvoie une chaine contenant l'ensemble des trophées du musée. Exemple de chaine :
`System.out.println(musee.donnerGauloisDonateur());`

```
Les trophées du musée sont :
- un casque
- un bouclier
- un casque
- une épée
```

4. Dans la classe « KeskonrixGestion » :
   - Ecrire l'attribut inventaire qui stocke pour chaque donateur les équipements donnés au musée associés à leur nombre d'exemplaire. Les donateurs seront placés dans l'ordre alphabétique. Exemple :
     {asterix = { Equipement.CASQUE=2, Equipement.BOUCLIER=1}, obelix = {Equipement.EPEE=1}}

   - Entourer dans la classe « Personne » (classe mère de la classe « Gaulois ») ce qui est indispensable au bon fonctionnement de la TreeMap

   - Compléter la méthode *ajouterTrophee* afin de remplir la map inventaire.

   - Créer la méthode *donnerInventaire* qui retourne une chaine correspondant à l'inventaire du musée. Exemple de chaine :
     System.*out*.println(musee.donnerInventaire());

```
INVENTAIRE DU MUSEE

Astérix a donné :
- 1 x un bouclier
- 2 x un casque

Obélix a donné :
- 1 x une épée

Ordralfabétix a donné :
- 1 x un casque
```

III. Classe KeskonrixGestion

```
public class KeskonrixGestion implements GestionTrophee {
   //Les attributs
   //Les associations
   private [                    ] trophees
                  = new [                    ];

   private [                              ]
         inventaire = new [                    ];

   // Les vues sur l'association

   private [                    ] gauloisDonnateur
                  = [                    ];

   private [                    ] collectionTrophees
                  = [                    ];

   //Les méthodes
   public void ajouterTrophee(Gaulois proprietaire, Equipement trophee) {
[                                                  ]
[                                                  ]
[                                                  ]
[                                                  ]
[                                                  ]
[                                                  ]
[                                                  ]
[                                                  ]
[                                                  ]

   //partie concernant la map inventaire

[                                                  ]
[                                                  ]
```

```
    }

    public String lesTrophees(Gaulois proprietaire) {
        String chaine = "Les trophées de " + proprietaire.getNom() + " sont :\n";

        return chaine;
    }
```

```
    public String tousLesTrophees() {
        String chaine = "Tous les trophées du musée sont :\n";
```

```
    }

    //Les méthodes pour la compréhension des vues
    public String donnerGauloisDonateur() {
        String chaine = "";
        if (
```

```
) {
            chaine = "Les gaulois ayant donné au moins un trophée au musée
                      sont :\n";
            for (
```

```
) {
```

```
            }
        } else {
            chaine = "Aucun gaulois n'a fait don d'un trophée au musée.";
        }
        return chaine;
    }

    public String donnerTrophees() {
        String chaine = "";
        if (
```

```
) {
            chaine = "Les trophées du musée sont :\n";
            for (
```

```
) {
```

```
            }
        } else {
            chaine = "Il n'y a aucun trophée au musée.";
        }
        return chaine;
    }
```

```
   //méthode concernant la map inventaire

   public String donnerInventaire(){
      String chaine = "\nINVENTAIRE DU MUSEE\n";

      for (                                                    ) {

         chaine += "\n" +                    .getNom() + " a donné :\n";

                                                                  

                                                                  

         for (                                                    ) {

            chaine += "- " +                                      

             + " x " +                                            
         }
      }
      return chaine;
   }
}
```

IV. Classe Personne

```
public class Personne implements Comparable<Personne>{
   protected String nom;

   public Personne(String nom) {
      this.nom = nom;
   }

   public String getNom() {
      return nom;
   }

   public String toString() {
      return nom;
   }

   public int compareTo(Personnage personnage) {
      return nom.compareTo(personnage.nom);
   }

   public boolean equals(Object object) {
      if(object != null && object.getClass() == getClass()){
         Personnage personnage = (Personnage) object;
         return nom.equals(personnage.nom);
      }
      return false;
   }
```

//méthode manquante pour l'utilisation comme clé dans une HashMap

}

V.Extrait de la JavaDoc TreeMap

Constructor and Description
TreeMap() Constructs a new, empty tree map, using the natural ordering of its keys.
TreeMap(**Comparator**<? super **K**> comparator) Constructs a new, empty tree map, ordered according to the given comparator.

Modifier and Type	Method and Description
boolean	**containsKey**(**Object** key) Returns true if this map contains a mapping for the specified key.
Boolean	**containsValue**(**Object** value) Returns true if this map maps one or more keys to the specified value.
V	**get**(**Object** key) Returns the value to which the specified key is mapped, or null if this map contains no mapping for the key.
Set<**K**>	**keySet**() Returns a **Set** view of the keys contained in this map.
V	**put**(**K** key, **V** value) Associates the specified value with the specified key in this map.
Collection<**V**>	**values**() Returns a **Collection** view of the values contained in this map.