جامعة هواري بومدين
للعلوم والتكنولوجيا
U S T H B

Corrigé d'exercices de TD POO

Série n°4 – Héritage et Polymorphisme

S. BOUKHEDOUMA

USTHB – FEI – département d'Informatique
Laboratoire des Systèmes Informatiques -LSI
sboukhedouma@usthb.dz

Exercice 5 –Série n°4

Exercice 5

On dispose d'une interface **TAB** définie par la donnée de la signature d'une méthode Fusion destinée à effectuer la fusion de deux vecteurs.

```
public interface TAB
{
  Public void Fusion (VECT V2, VECT V);
} // fin de l'interface TAB
```

On dispose aussi d'une classe **VECT** pour manipuler des vecteurs d'entiers définie comme suit:

Exercice 5 –Série n°4

```
public class VECT

{ private int t[ ] ; private int taille ;

  public VECT (int n) {taille = n ; t = new int [taille];} /* création du vecteur */

   public void saisir()
    {java.util.Scanner e = new java.util. Scanner (System.in);
     for (i=0 ; i<taille ; i++) t[i] = e.nextInt(); }

 public void afficher()
    {for (i=0 ; i<taille ; i++) ; System.out.print (t[i]+" ");}

} // fin de la classe VECT
```

Exercice 5 –Série n°4

Questions

1. Redéfinir les méthodes toString et equals dans la classe VECT

2. A partir de l'interface TAB et de la classe VECT, donner l'implémentation d'une classe **VectOrd** permettant de créer, saisir, afficher un vecteur trié d'entiers et fusionner deux vecteurs triés dans l'ordre croissant.

3. Ecrire le programme qui crée deux vecteurs triés, les fusionne et affiche le résultat de la fusion.

Exercice 3 –Série n°3

```
                    // suite de la classe VECT

public String toString ()      // redéfinition de la méthode toString
{ String S;
   for (int i =0; i<taille; i++ )
      { S = S + Integer.toString(t[i]);
         if (i<taille-1)   S = S + "\t";}
    return S;
```

Exercice 5 –Série n°4

```
                    // suite de la classe VECT

public boolean equals (Object obj)      // redéfinition de la méthode equals
{{if (this == obj)     return true;
   if (obj == null)    return false;
   if (this. getClass() != obj.getClass())
                       return false;
    VECT V= (VECT) obj;  // cast explicite

    if (V. taille!= this.taille) return false;

      // vérifier l'égalité des éléments deux à deux

int i =0;
while (i <this.taille && V.t[i] ==this.t[i])      { i++;}

if (i < this.taille)  return false;  else  return true;
} // fin de la classe VECT
```

Exercice 5 –Série n°4

Définition de la classe VectOrd à partir de la classe VECT et de l'interface TAB

```
public class VectOrd extends VECT implements TAB
{                                   // constructeur
  public  VectOrd ( int n)
       { super (n); }

            // redéfinition de la méthode saisir pour avoir des valeurs ordonnées

 public void Saisir()
   {java.util.Scanner e = new java.util. Scanner (System.in);
    System.out.print ("donnez une valeur");   t[0] = e.nextInt();
     for (int i=1 ; i<taille ;)
            { System.out.print ("donnez une valeur"); x = e.nextInt();
              if (x < t[i-1]) System.out.print ("donnez une valeur plus grande")
                 else {t[i] = x; i++; }  }
   }
```

Exercice 5 –Série n°4

Définition de la classe VectOrd à partir de la classe VECT et de l'interface TAB

```
// suite de la classe VectOrd
// implémentation de la méthode fusion de l'interface TAB
public void fusion( VECT V2,  VECT V)
   {int i, j, k;
    i = j = k =0;
      while (i<this.taille && j < V2.taille)
            {
             if (this.t[i] < V2.t[j])
                    { V.t[k] = this.t[i];  i++; k++;}
              else if (this.t[i] > V2.t[j])
                     { V.t[k] = V2.t[j];  j++; k++; }
               else { V.t[k] = V2.t[j];  j++; k++; V.t[k] = this.t[i];  i++; k++; }
            }
```

Exercice 5 –Série n°4

Définition de la classe VectOrd à partir de la classe VECT et de l'interface TAB

```
// suite de la méthode fusion


// éléments restants

        while (i<this.taille )
                { V.t[k] = this.t[i];  i++; k++;}

         while (j<V2.taille )
                 { V.t[k] = V2.t[j];  j++; k++;}
}
} // fin de la classe VectOrd
```

Exercice 5 –Série n°4

Définition de la classe VectOrd à partir de la classe VECT et de l'interface TAB

```
import java.util.Scanner;

class ProgVect
{  public static void main ( String args[])
    {  Scanner e = new Scanner (System.in);

                    // création des objets VectOrd
       System.out.println ("donner les tailles des vecteurs");
       int n1 = e.nextInt();  int n2 = e.nextInt();
        VectOrd  V1 = new VectOrd (n1);
        VectOrd  V2 = new VectOrd (n2);
        System.out.println ("remplissage de V1");  V1.saisir();
        System.out.println ("remplissage de V2");  V2.saisir();
                                        // méthode saisir de la classe VectOrd
```

Exercice 5 –Série n°4

Définition de la classe VectOrd à partir de la classe VECT et de l'interface TAB

```
                    // fusion des deux vecteurs

  VectOrd  V = new VectOrd (n1+ n2);  // Création d'espace pour le vecteur V
  V1.fusion(V2, V);
                                        // affichage du résultat

      System.out.println ("Voici le vecteur fusion");  V.afficher();

// ou bien
      System.out.println (V.toString());     // ou  System.out.println (V);
```