جامعة هواري بومدين
للعلوم والتكنولوجيا
U S T H B

Corrigé d'exercices de TD POO

Série n°2 – Classes et Objets

S. BOUKHEDOUMA

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

Exercice 1 –Série n°2

Exercice 1

1. Donner l'implémentation d'une classe **Vecteur** d'entiers définie par deux attributs, une référence et une taille. La classe doit comporter un constructeur, une méthode *remplir*, une méthode *afficher* et une méthode *somme* qui calcule la somme de deux vecteurs.

2. Ecrire un programme qui crée deux vecteurs, calcule leur somme et affiche le résultat.

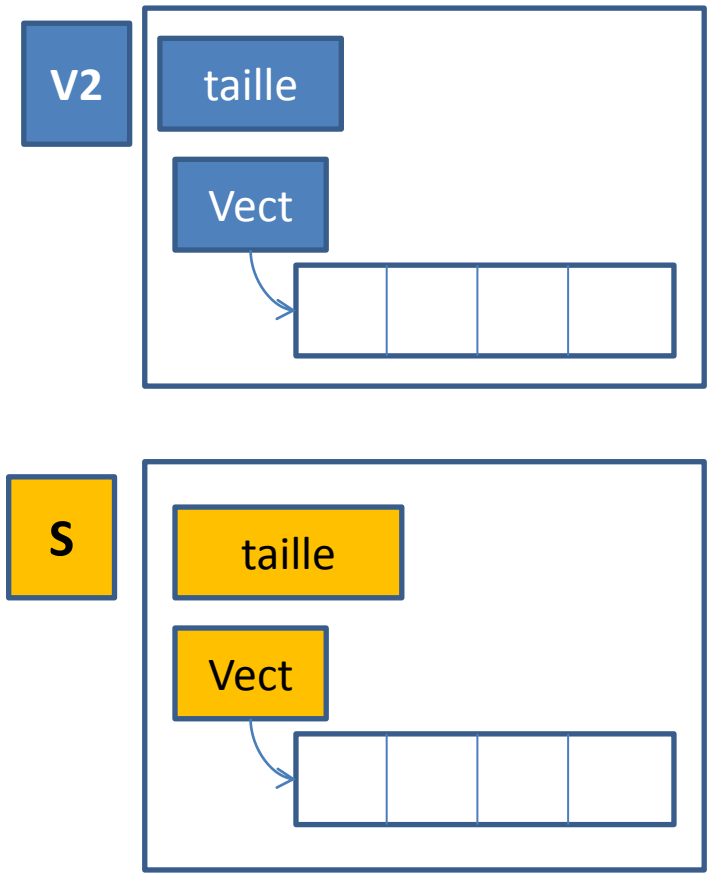
Exercice 1 –Série n°2

```
class Vecteur

{  int Vect[];   // référence
   int taille;

                        //constructeur
Vecteur (int n)
{ Vect = new int[n]; taille = n;}

                        //méthodes
void remplir ()
  {Scanner e = new Scanner(System.in);
  for (int i = 0; i<taille; i++)
      Vect[i] = e.nextInt(); }

void afficher ()
  {  for (int i = 0; i<taille; i++)
      System.out.print(Vect[i] + '\t'); }
```

/*Un vecteur est défini par une référence et une taille */

Exercice 1 –Série n°2

```
Vecteur Somme (Vecteur V2)

  {Vecteur S = new Vecteur (taille);
    for (int i = 0; i<this.taille; i++)
      S.Vect[i] = this.Vect[i]+V2.Vect[i];
     return S;}

//on peut écrire la méthode Somme autrement

void Somme (Vecteur V2, Vecteur S)
  {   for (int i = 0; i<this.taille; i++)
      S.Vect[i] = this.Vect[i] +V2.Vect[i]; }

} // fin de la classe Vecteur
```



Exercice 1 –Série n°2

class **ProgVecteur**

```
{  public static void main ( String args[])
    { // création de deux objets de type Vecteur

      Vecteur V1 = new Vecteur (20);
      Vecteur V2 = new Vecteur (20);
      System.out.println ("remplir V1");        V1.remplir();
       System.out.println ("remplir V2");       V2. remplir();

     // calcul du vecteur  somme en utilisant la 1ère méthode
       Vecteur S = V1.somme(V2);
        System.out.println ("le vecteur somme est:");   S.afficher();

       // Si on veut utiliser la 2ème méthode somme
         Vecteur S = new Vecteur (20);
         V1.somme (V2, S);
        System.out.println ("le vecteur somme est:");   S.afficher(); }
}
```

Exercice 2 –Série n°2

Exercice 2

On considère une entité **Pile** statique (sous forme d'un vecteur) d'entiers.

1. Implémenter une classe qui gère l'entité **Pile** comportant toutes les primitives de manipulation de la structure Pile.
2. Ecrire un programme qui crée une pile de *n* éléments et supprime toutes les répétitions d'une valeur val donnée.

Exercice 2 –Série n°2

```
class Pile

{  int  T [ ];   // référence de tableau
   int taille;   int  sommet;

                        //constructeur
Pile (int n)
{ T = new int [n]; taille = n;}

                    //méthodes -primitives
void initPile()
  {sommet = -1;}

void empiler (int x)
  {  sommet++;  T [sommet] = x;}
```

/*Une pile est représenté comme un vecteur avec un attribut en plus, le sommet */

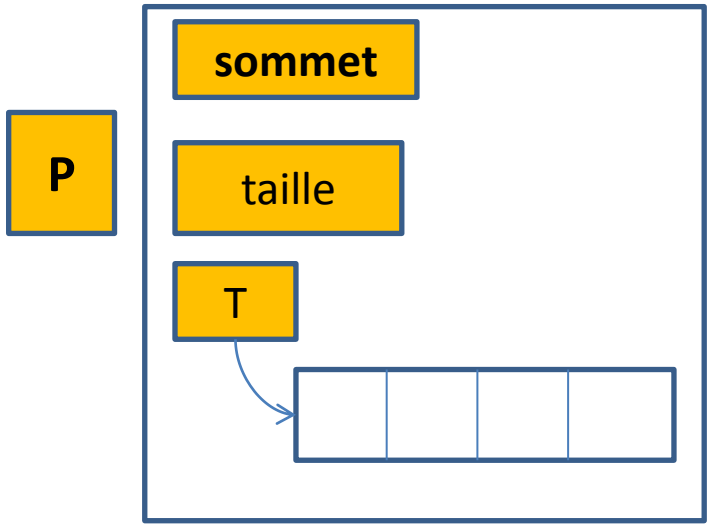
/*Le sommet est l'indice du vecteur, où se trouve la dernière valeur introduite et qui sera la 1ère retirée*/

Exercice 2 –Série n°2

```
int dépiler ()
{  int   x = T [sommet];
    sommet - -;  return x;}

int sommetPile ()
{ return (T [sommet]; }

boolean pileVide ()
  {return (sommet = = -1);}

boolean pilePleine ()
  {  return (sommet == taille-1);}

} // fin de la classe Pile
```

/*Schématiquement lorsqu'on crée un objet de type Pile, on aura en mémoire */

// Pile P = new Pile (10);



Exercice 2 –Série n°2

```
import java.util.Scanner;
class ProgPile

{  public static void main ( String args[])
     { Scanner  e = new Scanner (System.in);
       System.out.println ("donner la taille de la pile");
       int n = e.nextInt();

     // création d'un objet de type Pile
        Pile P = new Pile(n);

       // remplissage de la pile

       for (int i = 0; i<n; i++)
           {  System.out.println ("saisir une valeur");
              P. empiler (e.nextInt()); }
```

/*On appelle la méthode empiler sur la référence d'objet P de type Pile*/

Exercice 2 –Série n°2

```
//suppression des répétitions de val
Pile R = new Pile (n); R.initPile();   // Pile intermédiaire

System.out.println ("donner la valeur val");
       int val = e.nextInt();

While (! P.pileVide())
      { x= P.depiler();  R.empiler(x);    // ou bien  R.empiler (P.dépiler());
        if (x == val) break;}

If  (P.pileVide() && x!=val)
          System.out.println ("aucune occurrence de val dans la pile");
  else While (! P.pileVide())
      { x= P.depiler();  if (x != val)  R.empiler(x); }

           // remettre  les éléments de R dans P
While (! R.pileVide())
      { P.empiler (R.depiler()) };
}}
```

Exercice 3 –Série n°2

Exercice 3

On considère une entité Ouvrage décrite par une référence, un titre, un nom d'auteur et une année d'édition.

1. Donner l'implémentation de la classe Ouvrage, contenant un constructeur avec paramètres, une méthode de saisie des attributs, une méthode pour afficher la description complète d'un ouvrage et une méthode getAnnée pour retourner l'année d'édition de l'ouvrage.

2. Ecrire un programme java permettant de stocker la description de *n* ouvrages (entrés au clavier) dans un vecteur T de type Ouvrage.

3. Ecrire un programme java qui affiche la description de tous les ouvrages du vecteur T édités entre deux années données A1 et A2.

Exercice 3 –Série n°2

```
public class Ouvrage

{  private String ref, titre, auteur;
    private  int année;
                                        //constructeurs
public Ouvrage (String ref, String titre, String aut, int a)
{ this.ref= ref; this.titre = titre; auteur = aut; année = a; }
public Ouvrage () {}
                                        //méthodes
public void saisir ()
  {Scanner e = new Scanner(System.in);
    ref = e.nextLine(); titre = e.nextLine(); auteur = e.nextLine();
    année = e.nextInt();}

public void afficher ()
  {  System.out.print( ref+ '\t'+ titre+ '\t'+ auteur + '\t' + année ); }

public int getAnnée ()  {return année;}   // accesseur en lecture

}} //fin de la classe Ouvrage
```

Exercice 3 –Série n°2

```
class ProgOuvrage

{  public static void main ( String args[])
     {Scanner  e = new Scanner (System.in);
       System.out.println ("donner  le nombre d'ouvrages");
       int n = e.nextInt();

   // création du vecteur de références d'objets
  Ouvrage  V [ ] = new Ouvrage [n];  //initialisé à null

// création des objets Ouvrage

  for (int i = 0; i< V.length; i++)
          {   V[i] = new Ouvrage (); //constructeur  sans paramètres
              System.out.println ("saisir la description de l'ouvrage");
               V[i].saisir();}
```

Exercice 3 –Série n°2

```
// affichage des ouvrages édités entre deux années A1 et A2
System.out.println ("donner  les années A1 et A2);
        int  A1 = e.nextInt(); int  A2 = e.nextInt();
System.out.println ("les ouvrages édités entre " + A1 + "et " + A2);

    // parcours du vecteur
  for (int i = 0; i< V.length; i++)
          {   if (V[i].getAnnée () >= A1 && V[i].getAnnée () < = A2)
                  V[i].afficher();}

}} //fin de la classe ProgOuvrage
```

Remarque: *on accède à un attribut à l'extérieur de sa classe via un accesseur (get en lecture) pour respecter le principe d'encapsulation.*