ĐẠI HỌC ĐÀ NẴNG
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG VIỆT - HÀN
Vietnam - Korea University of Information and Communication Technology

OBJECT ORIENTED PROGRAMMING PRACTICE WITH JAVA

HUỲNH CÔNG PHÁP, Assoc. Prof, PhD.

Danang, 2021

Table of Contents

Chapter 1. Introduction to Java and Eclipse

I. Exercises with solutions

1. Install Eclipse

   https://www.tutorialspoint.com/eclipse/eclipse_installation.htm

2. Write a java program pirinting out: “Hello VKU freshmen!”

```
1
2 public class DataPrinting {
3
4     public static void main(String[] args) {
5
6         System.out.println("Hello VKU freshmen");
7
8     }
9 }
```

3. Write a java application allowing user to input a byte number N (32≤ N ≤255) and printing out its corresponding character.

```
1  import java.util.*;
2
3  public class Exercise2 {
4      public static void main(String[] args) {
5
6          byte ASCIIcode;
7          Scanner keyboard = new Scanner(System.in);
8          System.out.print("Input an ASCII code: ");
9          ASCIIcode = keyboard.nextByte();
10         System.out.println("ASCII code "+ ASCIIcode + " is " + (char)ASCIIcode)
11
12     }
13 }
14
```

```
Problems  Javadoc  Declaration  Console
<terminated> Exercise2 [Java Application] C:\Users\hcphap\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe  (13:44:25, 18 thg 9, 2021 – 13:44:28)
Input an ASCII code: 65
ASCII code 65 is A
```

4. Write a java application allowing user to input a character and printing out its corresponding ASCII code.

```java
import java.util.Scanner;
public class Exercise3 {
    public static void main(String[] args) {
        char ch;
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Input a character: ");
        ch = keyboard.next().charAt(0);
        System.out.println("ASCII code of "+ ch + " is " + (byte)ch);
    }
}
```

```
Problems  Javadoc  Declaration  Console
erminated> Exercise3 [Java Application] C:\Users\hcphap\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe  (13:57:16, 10 thg 9, 2021 – 13:57:20)
nput a character: a
SCII code of a is 97
```

5. Write a java application allowing user to input two integer numbers (*a* and *b*) then print out results of the following expressions: a+b; a-b; a*b; a/b; a%b

```java
import java.util.*;
public class Exercise4 {
    public static void main(String[] args) {
        Scanner k=new Scanner(System.in);
        System.out.println("Input two int numbers: ");
        int a=k.nextInt();
        int b=k.nextInt();
        System.out.println(a+" + "+b+" = "+(a+b));
        System.out.println(a+" - "+b+" = "+(a-b));
        System.out.println(a+" x "+b+" = "+a*b);
        System.out.println(a+" / "+b+" = "+a/b);
        System.out.println(a+" mod "+b+" = "+a%b);
    }
}
```

```
Problems  Javadoc  Declaration  Console
<terminated> Exercise4 [Java Application] C:\Users\hcphap\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-114
Input two int numbers:
3
6
3 + 6 = 9
3 - 6 = -3
3 x 6 = 18
3 / 6 = 0
3 mod 6 = 3
```

6. Write a java program allowing user to input a circle radius and calculating the circle's area and perimeter.

```java
import java.util.Scanner;
public class Exercise5 {

  public static void main(String[] args) {
    double r;
    final float pi=3.1415f;
    Scanner k=new Scanner(System.in);
    System.out.print("Input radius r= ");
    r=k.nextDouble();
    System.out.println("Circle area =  "+r*r*pi);
    System.out.println("Circle perimeter =  "+2*r*pi);
    }
}
```

Problems Javadoc Declaration Console

<terminated> Exercise5 [Java Application] C:\Users\hcphap\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-

```
Input radius r= 5
Circle area =  78.53749990463257
Circle perimeter =  31.414999961853027
```

II. Do it yourself

1. Write a java application allowing user to input a number (n) then print out results of the following expressions: sin(n), cos(n), $\sqrt{n}$

   Hint: Use the Math class

2. Write a java application allowing user to input a string and printing out the number of characters of the given string.

   Hint: Use the String class and length method

3. Write a java program allowing user to input width and high of a rectangle and calculating the rectangle's area and perimeter.

4. Explorer the online document
   https://www.codecademy.com/learn/learn-java

Chapter 2. Basic Object Oriented Programming Concepts

I. Exercises with solutions

1. Study and use class Math (java.lang.Math)

   Test at least 10 methods of the class Math, such as sin, cos, sqrt, ceil, round, min, max,....

```
public class Exercise1 {
     public static void main(String[] args) {
      int a =10, b =20 , c=30;
      int max;

      max = Math.max(a, b);
      System.out.println("Max of a, b is " + max);

      max = Math.max(max, c);
      System.out.println("Max of a, b, c is " + max);

      //put your code for other methods from here
      //......
     }
}
```

2. Study and use class String (java.lang.String). Test at least 10 methods of the class String.

```
public class Exercise2 {
     public static void main(String[] args) {
      String str1 = new String("VKU is one of the best
                               universities in Vietnam");
      String str2 = new String("Prof. Phap is an OOP
                                teacher");

      System.out.println("str1 contains "+
                          str1.length()+" character");

      int n=str1.compareTo(str2);
      if (n>0)
          System.out.println("str1 is greater than
                                    str2");
      else
      if (n==0)
       System.out.println("str1 is equal to str2");
      else System.out.println("str1 is less than str2");

      //put your code for other methods from here
      //......
```

```
    }
}
```

3. Study and use class Calendar (java.util.Calendar)

```
import java.util.*;
public class Exercise3 {
     public static void main(String[] args) {

          Calendar rightNow = Calendar.getInstance();
          System.out.println(rightNow.getTime());

          System.out.println(rightNow.getTimeZone());

          System.out.println(rightNow.getWeekYear());

          //put your code for other methods from here
          //......

     }
}
```

4. Study and use class Date (`java.util.Date`)

```
import java.util.*;

public class Exercise4
{
    public static void main(String[] args)
    {
        Date d1 = new Date();
        System.out.println("Current date is " + d1);
        System.out.println("Current hour is
                           "+d1.getHours());

        //put your code for other methods from here
        //......

    }
}
```

5. Study and use class LocalDate (java.time.LocalDate)

```
import java.time.LocalDate; // import the LocalDate class

public class Exercise4 {
  public static void main(String[] args) {
    LocalDate myObj = LocalDate.now();
    System.out.println(myObj);
    //put your code for other methods from here
    //......

  }
}
```

6. Specify and implement an OOP program to calculate the area of a rectangle.

+ Specification:

class Rectangle
properties: double width, height
method: area( ) width * height

+ Implementation:

```
class Rectangle
{
   private double width, height;
   Rectangle(double widVal,double heightVal)
        {

           width = widVal;
           height = heightVal;

        }
    double area()
      {
           return width * height;
      }
}

public class Exercise6
{
  public static void main(String args[])
   {
   Rectangle rec1, rec2;
```

```
    rec1 = new Rectangle(23,20);
    rec2 = new Rectangle(40,50);
    System.out.println("area of rec1 is : " +
                        rec1.area());
    System.out.println("area of rec2 is : " +
                        rec2.area());
        }
}
```

```
area of rec1 is : 460.0
area of rec2 is : 2000.0
```

7. Specify and implement an OOP program to calculate the cube volume.

+ Specification:

class Cube
properties: double size
methods: + setSize(val) size=val + getSize() get size value + volume( ) size * size* size + details( ) display detailed infos

+ Implementation:

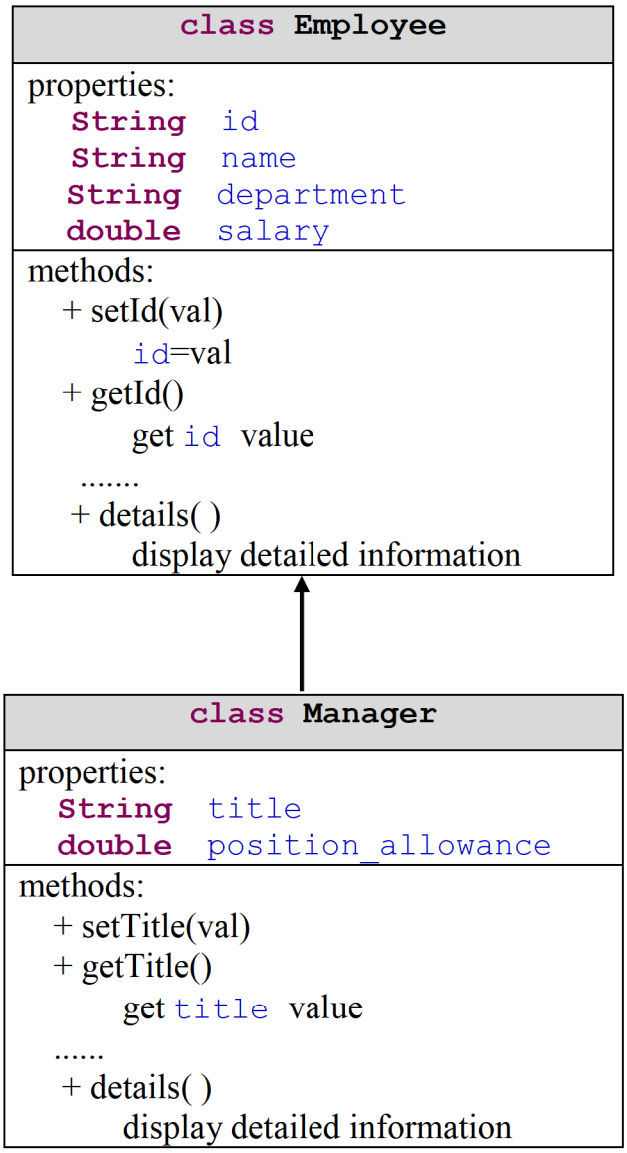
```
class Cube
{
   private double size;
   Cube(double val)
     {
          size = val;
     }

   void setSize(double val)
    {
      size = val;
    }
```

```
    double getSize()
     {
      return size;
     }
    double volume()
     {
     return size*size*size;
     }
    void details()
     {
     System.out.println("\ndetails of rectangle");
     System.out.println("width="+size);
     System.out.println("height="+size);
     System.out.println("length="+size);
     System.out.println("volume="+volume()+"\n\n");
    }
}
public class Exercise7
{
 public static void main(String args[])
    {
     Cube cube1, cube2;
     cube1 = new Cube(4);
     cube2 = new Cube(3);

     System.out.println("cube1's size= "+cube1.getSize());
     System.out.println("cube2's size= "+cube2.getSize());

     cube1.setSize(10);
     cube2.setSize(20);

      cube1.details();
      cube2.details();
    }
}
```

```
cube1's size= 4.0
cube2's size= 3.0

details of rectangle
width=10.0
height=10.0
length=10.0
volume=1000.0


details of rectangle
width=20.0
height=20.0
length=20.0
volume=8000.0
```

8. Specify and implement an OOP program to create a class Employee and a class Manager inheriting from the class Employee.

+ Specification:



+ Implementation:

```
class Employee
{
     String     id;
     String     name;
     String     department;
     double     salary;
public Employee(String id, String name, String dep,
                 double sal)
    {
      this.id=id;
      this.name=name;
      department=dep;
      salary =sal;
    }
public void setId(String val)
    {
      id=val;
    }
public String getId()
    {
      return id;
    }
//complete code for getX, setX of other properties
public void details()
    {
      System.out.println("ID: "+id);
      System.out.println("Name: "+name);
      System.out.println("Department: "+department);
      System.out.println("Salary: "+salary);
    }
}

class Manager extends Employee
    {
      String     title;
      double     position_allowance;

public Manager(String id, String name, String dep,
                double sal, String tit, double pa)
    {
      super(id,name,dep,sal);
      title=tit;
      position_allowance=pa;
    }
public void setTitle(String val)
    {
      title=val;
    }
```

```
public String getTitle()
   {
      return title;
   }
//complete code for getX, setX of other properties
public void details()
   {
   super.details();
   System.out.println("Title: "+title);
   System.out.println("Position allowance: "
                        + position_allowance);
}
}
public class Exercise8 {

public static void main(String[] args) {
 Employee A = new Employee("VKU01",
           "David Tho","Academic Office",10000);
 Manager  B = new Manager("VKU02",
     "Jonh Phap","Steering Board",10000,"Rector",1000);
   A.details();
   System.out.println("--------");
   B.details();
}
}
```

```
ID: VKU01
Name: David Tho
Department: Academic Office
Salary: 10000.0
--------
ID: VKU02
Name: Jonh Phap
Department: Steering Board
Salary: 10000.0
Title: Rector
Position allowance: 1000.0
```

II. Do it yourself

1. Study and use class Character (java.lang. Character). Test at least 10 methods of this class.
2. Study and use class StringBuffer (java.lang. StringBuffer). Test at least 10 methods of this class
3. Study and use class Random (java.util. Random). Test at least 10 methods of this class

   Hint: import java.util. Random;

4. Study and use other classes in packages java.lang and java.util. For each class, test at least 05 methods.

5. Specify and implement an OOP program to calculate average mark and display information of the student.

   Hint: student information includes: name, age, address, marks of subjects,...

6. Specify and implement an OOP program to calculate benefit and display information of the company.

   Hint: benefit = income – cost
   Company information includes name, address, cost, income, benefit...

7. Specify and implement an OOP program to create a class Person and a class Teacher inheriting from the class Person.

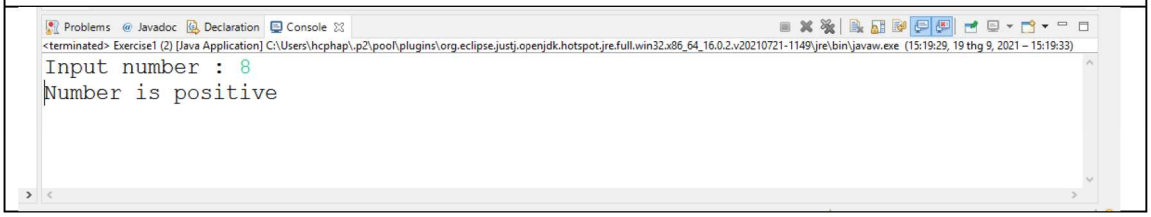
   Hint: Person (name, age, address)
   Teacher(..., institution_name, courses,...)

Chapter 3. Java fundamentals and control structures

I. Exercises with solutions

1. Write an OOP program to get a integer from the keyboard and check if it positive, negative or zero?

```
import java.util.Scanner;
public class Exercise1 {
  public static void main(String[] args) {
    Scanner k=new Scanner(System.in);
    System.out.print("Input number : ");
    int n=k.nextInt();
    if(n==0)System.out.print("Zero");
    else
      if(n>0) System.out.print("Number is positive");
      else System.out.print("Number is negative");
 }

}
```



2. Write an OOP program to solve a quadratic equation (ax2 + bx + c =0)

```
import java.util.Scanner;

public class QuadractiEquation {
  public static void main(String[] args) {

  float a,b,c,x1,x2,del;

  Scanner keyboard = new Scanner(System.in);

  System.out.print("a = ");
  a = keyboard.nextFloat();

  System.out.print("b = ");
  b = keyboard.nextFloat();

  System.out.print("c = ");
  c = keyboard.nextFloat();

     del = b*b - 4*a*c;
     if (del < 0) System.out.println("Roots are complex
                                      and different");
     else
```

```
        if (del==0)
          {
          x1=x2=-b/(2*a);
          System.out.println("Roots are real and same: "+x1);
          }
        else //means del > 0
           {
           x1=(float)(-b+Math.sqrt(del))/(2*a);
           x2=(float)(-b-Math.sqrt(del))/(2*a);
           System.out.println("Roots are real and different:
                               \n x1="+x1+"\n x2="+x2);
           }
      }
}
```

```
a = 3
b = 7
c = 2
oots are real and different:
 x1=-0.33333334
 x2=-2.0
```

Requirement: Students must complete the code above for complex roots

3. Write an OOP program allowing users to input three numbers (a, b, c) from keyboard. Check if a, b, c numbers are 3 edges of a triangle? If yes, print out the type of triangle?

```
import java.util.Scanner;

public class TriangleCheck {

 public static void main(String[] args) {

  float a,b,c;

  //nhập dữ liệu từ bàn phím
  Scanner keyboard = new Scanner(System.in);
  System.out.print("a = ");
  a = keyboard.nextFloat();

  System.out.print("b = ");
  b = keyboard.nextFloat();

  System.out.print("c = ");
  c = keyboard.nextFloat();

/*Check if 3 triangle edges are valid*/
```

```
  if ((a+b>c) && (a+c>b) && (b+c>a)&&(a>0) && (b>0)&&(c> 0))
   {
   System.out.print("a, b, c are valid \n");

/*Get type of the triangle*/
     if ((a==b)&&(b==c)) System.out.println("Equilateral
triangle");
      else
     if((a==b)||(b==c)||(a==c))
System.out.println("Isosceles triangle");
      else
       if
(((a*a+b*b==c*c)&&(a==b))||((a*a+c*c==b*b)&&(a==c))
               || ((c*c+b*b==a*a)&&(c==b)))
                          System.out.println("Isosceles right
triangle");
       else
       if ((a*a==b*b+c*c)||(b*b==a*a+c*c)||(c*c==a*a+b*b))
         System.out.println("Right triangle");
       else
          System.out.println("Triangle");
     }
   else //belong to the first if
       System.out.println("\"a, b, c are NOT valid \"");
  }
}
```

4. Write an OOP program to sum all even numbers from 2 to N.

```
import java.util.Scanner;
public class SumEven
{
 public static void main(String[] args)
    {
      Scanner keyboard = new Scanner(System.in);
      System.out.print("N = ");
      int N = keyboard.nextInt();

     int sum =0;
     for (int i=1;i<=N;i+=1)
         if (i%2==0) sum+=i;

     System.out.println("Sum = "+sum);
     }
}
```

5. Write an OOP program to check whether N (N>0), which got from the keyboard, is a prime number or not?

```
import java.util.Scanner;
public class PrimeNumber {
 public static void main(String[] args)
 {
  int N;
  int i;

  Scanner keyboard = new Scanner(System.in);
  System.out.print("N = ");
  N = keyboard.nextInt();

/*Kiểm tra số nguyên tố*/

  for (i=2;i<=Math.round(Math.sqrt(N));i++)
    if (N%i==0) break;/*Nếu chia hết cho một số i thì N
                        không phải số nguyên tố*/

/*Nếu i nhỏ hơn hoặc bằng căn bậc 2 N có nghĩa vòng lặp bị kết
thúc bằng câu lệnh break*/
  if (i <= Math.round(Math.sqrt(N)))
    System.out.print("N is not a prime number ");
  else System.out.print("N is a prime number ");
 }

}
```

```
Problems  Javadoc  Declaration  Console
<terminated> PrimeNumber [Java Application] C:\Users\hcphap\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe (15:53:30, 19 thg 9, 2021 – 15:53:34)
N = 7
N is a prime number
```

II. Do it yourself

1. Write an application that prints all of integer the numbers between 1 (included) and 100 (included) using the "while" loop.
2. Write an application that prints all of the integer numbers between 1 (included) and 100 (included) using the "do..while" loop.
3. Write an application that prints all of the integer numbers between 1 (included) and 100 (included) using the "for" loop.
4. Write an application that prints all of the even numbers between 1 (included) and 100 (included) using the "while" loop.

5. Write an application that prints all of the even numbers between 1 (included) and 100 (included) using the "do..while" loop.
6. Write an application that prints all of the even numbers between 1 (included) and 100 (included) using the "for" loop.
7. Write an application that calculates the sum of all the integer numbers between 1 (included) and 100 (included).
8. Write an application that calculates and prints the average of all the integer numbers between 1 (included) and 100 (included).
9. Write an application that calculates and prints the biggest number between 1 (included) and 100 (included) that divides in 7 without a residual.
10. Write an application that calculates the sum of all the numbers between 1 (included) and 100 (included) that divide in 7 without a residual.
11. Write an application that prints all of the integer numbers between 1 (included) and 1000 (included) and near each one of them (each number will be printed in a new line) it prints EVEN or UNEVEN.
12. Write an application that prints a rectangle of stars as follows:

```
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
* * * * * * * * * *
```

13. Write an application that prints a shape of stars as follows:

```
*  *  *  *  *  *  *  *  *  *
   *  *  *  *  *  *  *  *
      *  *  *  *  *  *
         *  *  *  *
      *  *  *  *  *  *
   *  *  *  *  *  *  *  *
*  *  *  *  *  *  *  *  *  *
```

14. Write an application that prints to the screen the following:

```
            *
         *  *  *
      *  *  *  *  *
*     *  *  *  *  *  *     *
```

15. Write an application that prints to the screen the following:

```
            *
         *  *  *
      *  *  *  *  *
*     *  *  *  *  *  *     *
      *  *  *  *  *
```

```
*     *     *
      *
```

16. Write an application that prints to the screen the following series:

    0, 3, 8, 15, 24, 35, 48, . . .

17. Write an application that prints to the screen the following series:

    1, 3, 7, 15, 31, 63, . . .

18. Write an application that prints to the screen the following series:

    1, 7, 16, 37, 79, 173 ….

    The application should print the first 10 numbers of the series.

19. Write an application that prints to the screen the following series:

    1, 3, 9, 27, 81, 243 ….

    The application should print the first 10 numbers of the series.

20. The following application prints the factorial of 6. The computation of 6! is done using a static method that calculates the factorial of the number that it gets. Complete the missing lines.

```
class FactorialApplication
{
        public static void main(String args[])
        {
                long number, result;
                number = 6;
                result = factorial(number);
                System.out.println("The factorial of 6 is : " + result);
        }
        public static long factorial(long value)
        {
                long result = 1;

                //add the missing lines here
                return result;
        }
}
```

21. Develop a stand-alone application that prints each one of the numbers between 0 and FFFF (the numbers should be printed in Hexadecimal base).

22. Develop a stand-alone application that prints each one of the numbers between 0 and 777 (the numbers should be printed in Octal base).

Chapter 4. Java methods, arrays and references

I. Exercises with solutions

1. Write an OOP program including a class that describes a point in the 3D space (each object of this class should have three variables: x, y and z). The class should have the following instance methods: a method that calculates the distance between the current point (on which the method in invoked) to another point; a method that calculates the distance between the current point (on which the method is invoked) to the center. a method that prints all the details of the point.

```
public class Point3D
{
     private double x,y,z;
     public Point3D(double x, double y, double z)
      {
      this.x = x;
      this.y = y;
      this.z = z;
      }
    public double distanceFromCenter()
      {
      return distanceFrom(0,0,0);
      }
    public double distanceFrom(Point3D other)
      {
      return Math.sqrt(Math.pow(x-other.x,2)+ Math.pow(y-
other.y,2)
                        + Math.pow(z-other.z,2));
      }
    public double distanceFrom(double xVal, double yVal,
double zVal)
     {
     return Math.sqrt(Math.pow(x-xVal,2)+
                      Math.pow(y-yVal,2)+
                      Math.pow(z-zVal,2));
       }
    public void details()
       {
        System.out.println("x="+x+ "y="+y+" z="+z);
       }
    public static void main(String args[])
      {
        Point3D point1, point2;
        point1 = new Point3D(2,3,4);
        point2 = new Point3D(3,7,8);
```

```
        System.out.print("point1:");
        point1.details();
        System.out.print("point2:");
        point2.details();
        System.out.println("The distance between point1 and
(0,0,0) is : " +
                           point1.distanceFromCenter());
        System.out.println("The distance between point1 and
point2 is : " +
                           point1.distanceFrom(point2));
            }
        }
```

2. Write an OOP program sorting an array of integers in ascending order.

Solution 1:

```
public class ArraySort {
    public static void main(String[] args) {

/*Tạo và khởi tạo giá trị cho mảng*/
        int[] A = {5, 7, 2, 4, 8};

/*Sắp xếp mảng theo chiều tăng dần*/
        for (int i=0;i<A.length-1;i++)
           for (int j=i+1;j<A.length;j++)
            if (A[i]>A[j])
            {
                 int t =A[i];
                 A[i]=A[j];
                 A[j]=t;
            }

/*In mảng ra màn hình*/
        for (int i=0;i<A.length;i++)
        System.out.print(A[i]+" ");
    }
```

Solution 2.

```
/* Use class Arrays in java.util*/
import java.util.Arrays;

public class ArraySort {

    public static void main(String[] args) {

/*Tạo và khởi tạo giá trị cho mảng*/
```

```
        int[] A = {5, 7, 2, 4, 8};

/*Sử dụng hàm sort của lớp Arrays để sắp xếp mảng A theo chiều tăng
dần*/
        Arrays.sort(A);

/*Hiển thị kết quả ra màn hình*/
        for (int i=0;i<A.length;i++)
        System.out.print(A[i]+" ");
    }
    }
```

3. Write an application that creates a two dimensions array 10 X 10 and stores the multiplication table in it. The application should also prints the multiplication table values.

```
public class TwoDemenArray
{
          public static void main(String args[])
          {
                    int matrix[][];
                    matrix = new int[10][10];
                    for(int i=0;i<10;i++)
                    {
                     for(int j=0;j<10;j++)
                       {
                        matrix[i][j] = (i+1)*(j+1);
                       }
                    }
                    for(int i=0;i<10;i++)
                    {
                     for(int j=0;j<10;j++)
                        {
                        System.out.print(matrix[i][j]+"\t");
                        }
                        System.out.println();
                    }
          }

          }
```

4. Write an OOP program in which includes a factorial method. Calculating S = 7! + 10! + 12! + 14!

```
   public class Factorial {

/*Định nghĩa hàm tính giai thừa*/
      public static long fact(int n)
      {

        /*Tính n giai thừa*/
            int kq =1;
            for (int i=2;i<=n;i++)
                  kq *=i;
            return kq;
      }

      public static void main(String[] args) {

   /*Gọi hàm fact, do hàm fact là static nên không cần tạo đối
   tượng để gọi hàm*/
       long S = fact(7) + fact(10) + fact(12) + fact(14);

   //In kết quả ra màn hình
       System.out.println("Sum ="+S);

       }

   }
```

```
<terminated> Factorial [Java Application] C:\Program Files\Java\jdk1.7.0_45\jre\bin\javaw.exe (Nov 18, 2013, 11:02:17 AM)
Sum =1761580720
```

5. Write an OPP program to find the greatest common divisor of two integers entered from the keyboard.

```
   import java.util.Scanner;
   public class CommonDivisor {

   /*Định nghĩa hàm tính ước số chung lớn nhất*/
      public static int comDiv(int a,int b)
       {
            return (b==0)?a:comDiv(b, a%b);
       }
       public static void main(String[] args)
```

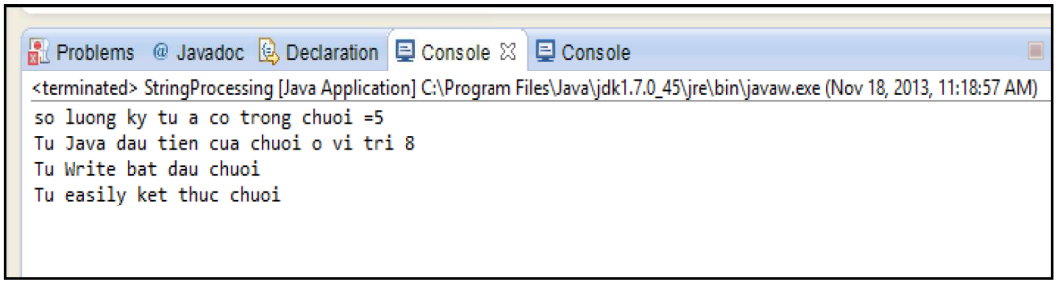
```
        {
         int a;
         int b;

    /*nhập dữ liệu từ bàn phím*/
         Scanner keyboard = new Scanner(System.in);
         System.out.print("a = ");
         a = keyboard.nextInt();

         System.out.print("b = ");
         b = keyboard.nextInt();

    /*Gọi hàm và in kết quả ra màn hình*/
         System.out.println("USCLN cua "+a+" va "+b+" la
    "+comDiv(a,b));
        }
    }
```

```
<terminated> CommonDivisor [Java Application] C:\Program Files\Java\jdk1
a = 16
b = 10
USCLN cua 16 va 10 la 2
```

6. Write an OOP program for processing a give string:
   (a) Count the frequency of character *a* in the string
   (b) Check if the string contains the word "Java"?
   (c) Check if the string starts with the word "Write"?
   (d) Check if the string ends with the word "easily"

```
public class StringProcessing {

 public static void main(String[] args) {
       String s ="Write a Java program very easily";

/*Dùng để chứa số lượng ký tự 'a' có trong chuỗi*/
       int count=0;

/* Đếm số ký tự 'a'*/
       for (int i=0;i<s.length();i++)
            if (s.charAt(i)=='a') count++;
       System.out.println("so luong ky tu a co trong chuoi ="+count);

  /*Kiểm tra chuỗi có chứa từ "Java" hay không*/
       int index =s.indexOf("Java");
```

```
      if (index>=0) System.out.println("Tu Java dau tien cua chuoi o
vi tri "+index);
          else System.out.println("Chuoi khong chua tu Java");

/*Kiểm tra chuỗi có bắt đầu bằng cụm từ "Write" */
     if (s.startsWith("Write")==true) System.out.println("Tu Write
bat dau chuoi");
      else System.out.println("Tu Write khong bat dau chuoi");

  /*Kiểm tra chuỗi có kết thúc bằng từ "easily" */
     if (s.endsWith("easily")==true) System.out.println("Tu easily
ket thuc chuoi");
      else System.out.println("Tu easily khong ket thuc chuoi");
 }
}
```



7. Write an OOP program, using the Vector class in java.util package to store fruits. Display its size, capacity and elements.

```
import java.util.*;
public class Reference {
   public static void main(String args[]) {
      /* Vector of initial capacity(size) of 6 */
      Vector<String> vec = new Vector<String>(6);

      /* Adding elements to a vector*/
      vec.addElement("Banana");
      vec.addElement("Watermelon");
      vec.addElement("Mango");
      vec.addElement("Apple");

      /* check size and capacityIncrement*/
      System.out.println("Size is: "+vec.size());
      System.out.println("Default capacity increment is:
"+vec.capacity());

      vec.addElement("Grape");
      vec.addElement("Orange");
      vec.addElement("Clipart");
```

```
        /*size and capacityIncrement after two insertions*/
        System.out.println("Size after addition: "+vec.size());
        System.out.println("Capacity after increment is:
"+vec.capacity());

        /*Display Vector elements*/
        Enumeration en = vec.elements();
        System.out.println("\nElements are:");
        while(en.hasMoreElements())
           System.out.print(en.nextElement() + " ");
    }
}
```

II. Do it yourself

1. Write an OOP program with a method to find the smallest number among three numbers.
2. Write an OOP program with a method to count all vowels in a string.
3. Write an OOP program with a method to count all words in a string.
4. Write an OOP program with a method to compute the sum of the digits in an integer.
5. Write an OOP application that stores the factorials of all the numbers between 1 (included) and 10 (included) in an array and prints them in a reverse order.
6. Write an OOP program sorting an array of integers in descending order.
7. Write an OOP program to check if the first and the last element of an array of integers are same.

   Hint. The length of the array must be greater than or equal to 2.
8. Write an OOP program to swap the first and last elements of an array (length must be at least 1) and create a new array.
9. Write an OOP program to multiply corresponding elements of two arrays of integers

   Hint. Sample Output:
   Array1: [2, 3, 5, 3]
   Array2: [2, 4, -5, -2]
   Result: 4 12 -25 -6
10. Write an OOP program to check if a string starts with a specified word enterd from the keyboard.
11. Write an OOP program to create a new array from a given array of integers (two array have the same size).

12. Write an OOP program to find the maximum and minimum value of an array.
13. Write an OOP program to reverse an array of integer values.
14. Write an OOP program to find the common elements between two arrays (string values)

    Hint.

```
Array1 : [Python, JAVA, PHP, C#, C++, SQL]
Array2 : [MySQL, SQL, SQLite, Oracle, PostgreSQL, DB2, JAVA]
Common element is : [JAVA, SQL]
```

15. Write an OOP program with a method to displays an 8-by-8 matrix.
16. Write an OOP program to convert an array to ArrayList.
17. Write an OOP program to compare two strings lexicographically. Two strings are lexicographically equal if they are the same length and contain the same characters in the same positions.
18. Write an OOP program to concatenate a given string to the end of another string
19. Write an OOP program to get a substring of a given string between two specified positions.
20. Write an OOP program to check if a given string contains the specified substring.
21. Write an OOP program to create a new String object with the contents of a character array.
22. Write a OOP program to check whether two String objects contain the same data.
23. Write an OOP program to count the occurrences of a given string in another given string.

    Hint.
    Sample output: 'aa' has occured 3 times in 'abcd abc aabc baa abcaa'
24. Write an OOP method to check whether a string is a valid password.
    - Password rules:
    - A password must have at least ten characters.
    - A password consists of only letters and digits.
    - A password must contain at least two digits.
25. Write an OOP program, using the Hashtable class in java.util package to store a list of ids and studentnames. Display its elements information.

    Hint.
    Import java.util.;*
    ...
    Hashtable list = new Hashtable();
    list.put("VKU01", "Nguyen Van A") ;
    list.put("VKU02", "Huynh Van B") ;

```
Enumeration elements = list.elements();
.....
```

Chapter 5. Implementation of Abstraction and Encapsulation

I. Exercises with solutions

1. Write an OOP program, declaring a class that describes a bank account. The class should have the following attributes balance, id and it should also have the following two methods deposit(double), withdraw(double) and others.

```
class Account
{
      private double balance;
      private long id;
          //constructor
      public Account(double balanceVal, long idVal)
      {
       balance = balanceVal;
       id = idVal;
      }
     public void withdraw(double sum)
      {
       balance -= sum;
      }
     public void deposit(double sum)
      {
      balance += sum;
      }
     public void details()
      {
      System.out.println("\nid="+id);
      System.out.println("balance-"+balance);
      }
}
public class BankAccount
{
     public static void main(String args[])
      {
       Account acc1;
       acc1 = new Account(100,123123);
       System.out.println("\ncreating new account");
       acc1.details();
       System.out.println("\ndeposit 2000");
       acc1.deposit(2000);
       acc1.details();
       System.out.println("\nwithdrawing 1000");
       acc1.withdraw(1000);
       acc1.details();
      }
}
```

```
Creating new account

id=123123
balance=100.0
Deposit 2000

id=123123
balance=2100.0
Withdrawing 1000

id=123123
balance=1100.0
```

2. Write an OOP program to create a class Point2D with methods to move a point to new position, to calculate the distance between two points:

Specification:

Attributes	Description
float x,y	Coordinates
Methods	**Description**
Point2D()	Default construction
Point2D(float, float)	Construction with two parametters
void move(float dx, float dy)	Move to a new position (x+dx, y+dy)
float distance(Point)	Distance between the current poit to another: $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$
void display()	Display a point's coordinates

```
public class Point2D {
/* Khai báo hai thuộc tính x, y */
      private float x, y;
/*Hàm khởi tạo mặc định*/
      public Point2D()
      {
            this.x=0;
            this.y=0;
      }
/*Hàm khởi tạo 2 thuộc tính*/
      public Point2D(float x, float y)
      {
            this.x=x;
            this.y=y;
      }
/*Thay đổi vị trí mới*/
      public void move(float dx, float dy)
      {
            x+=dx;
            x+=dy;
      }
/*Tính khoảng cách từ điểm hiện tại đến điểm A*/
```

```
      public double distance(Point2D A)
      {
 /*Sử dụng hàm sqrt và pow của lớp Math để tính căn bậc hai và
mũ*/
return Math.sqrt(Math.pow(this.x-A.x,2)+Math.pow(this.y-
A.y,2));
      }
/*Hiển thị tọa độ của điểm hiện tại*/
      public void display()
      {
            System.out.println("("+x+","+y+")");
      }
      public static void main(String args[])
      {
    /*Tạo 2 điểm A và B*/
            Point2D A,B;
            A=new Point2D(3,4);
            B=new Point2D(6,7);

   /*Hiển thị tọa độ của A và B*/
            System.out.print("A co toa do la "); A.display();
            System.out.print("B co toa do la "); B.display();

   /*Di chuyển đến vị trí mới*/
            A.move(4, 2);

   /*Hiển thị tọa độ điểm A ở vị trí mới*/
            System.out.print("Toa do moi cua A la ");
A.display();

   /*Tính khoảng cách từ điểm A đến điểm B*/
            double d = A.distance(B);
            System.out.println("Khoan cach tu A den B la "+d);

      }

}
```

```
Problems  @ Javadoc  Declaration  Console  Console
<terminated> Point2D [Java Application] C:\Program Files\Java\jdk1.7.0_45\jre\b
A co toa do la (3.0,4.0)
B co toa do la (6.0,7.0)
Toa do moi cua A la (9.0,4.0)
Khoan cach tu A den B la 4.242640687119285
```

3. Write an OOP program, creating a class Point class and a class Line. The class Line has methods length, isParallel and others. The length method should return the length of the line and the isParallel method should check whether this is parallel to an other object that describes a line.

```
class Point2D
{
   private double x;
   private double y;
   Point2D(double xVal, double yVal)
     {
      x = xVal;
      y = yVal;
     }
   public double getX()
     {
      return x;
     }
   public double getY()
     {
     return y;
     }
   public void details()
     {
     System.out.println("("+x+","+y+")");
     }
}
class Line2D
{
   private Point2D first;
   private Point2D second;
   public Line2D(Point2D p1, Point2D p2)
    {
    first = p1;
    second = p2;
    }
   public void details()
    {
    System.out.println("(("+first.getX() + "," +
first.getY() + ") , (" + second.getX() + "," +
second.getY() + "))");
    }
   public boolean isParallel(Line2D otherLine)
    {
     //add your lines here
    double thisDelta = (first.getY()-second.getY()) /
(first.getX()-second.getX());
```

```
    double otherDelta = (otherLine.first.getY()-
otherLine.second.getY()) / (otherLine.first.getX()-
otherLine.second.getX());
    return (thisDelta==otherDelta);
    }
   public double length()
    {
     //add your lines here
     return Math.sqrt( Math.pow(first.getX()-
second.getX(),2) + Math.pow(first.getY()-second.getY(),2)
);
    }
 }
public class LineTest
{
   public static void main(String args[])
    {
    Line2D line1, line2;
    line1 = new Line2D(new Point2D(10,20), new
Point2D(5,15));
    line2 = new Line2D(new Point2D(4,4), new
Point2D(33,22));
    System.out.println("The length of line1 is : " +
line1.length());
    System.out.println("The length of line2 is : " +
line2.length());
    if(line1.isParallel(line2))
      {
       System.out.println("line1 is parallel to line2");
      }
     else
      {
       System.out.println("line1 is not parallel to
line2");
      }
     }
}
```

```
The length of line1 is : 7.0710678118654755
The length of line2 is : 34.132096331752024
line1 is not parallel to line2
```

4. Write an OOP program, creating a class MyMath providing several static methods as follows:

- factorial, that receives one int value and returns its factorial. The returned value type is long.

- isPrime, that receives a number and check whether it is a prime number. The returned value type is boolean.
- isTriangle, that receives three numbers and check whether these numbers can create a triangle. The returned value type is boolean.

```
class MyMath
{
     public static long factorial(int num)
      {
       long result=1;
       for(int i=2;i<=num;i++)
        {
        result*=i;
        }
       return result;
       }
    public static boolean isPrime(long number)
     {
      long numberSqrt = (long)Math.sqrt(number);
      boolean result = true;
      for(int i=2; i<numberSqrt && result; i++)
       {
       if(number%i==0)
       result = false;
       }
      return result;
      }
   public static boolean isTriangle(double num1, double num2,
double num3)
     {
     return (num1>(num2+num3) && num2>(num1+num3) &&
num3>(num2+num1));
     }
}
public class MyUtilTest
{
   public static void main(String args[])
     {
     double a=12,b=2,c=8;
     System.out.println("Saying that 12, 2 and 8 create a
triangle is : "
                        + MyMath.isTriangle(12,2,8));
     System.out.println("The factorial of 6 is : "
                        + MyMath.factorial(6));
     System.out.println("Saying that 17 is a prime number is :
"
                        + MyMath.isPrime(17));
      }
```

```
}
```

```
Saying that 12, 2 and 8 create a triangle is : false
The factorial of 6 is : 720
Saying that 17 is a prime number is : true
```

5. Write an OOP program, creating a class Time containing attributes second, minute, hour and methods: constructions; get and set values for the attributes; change hour, minute, second. At the main method, create an Object of the class Time with hour, minute, second corressponding to 7,0,0, then display the clock.

```
   public class Time {

/* Khai báo 3 thuộc tính giờ, phút, giây*/
      private int hour;     // 0 - 23
      private int minute;   // 0 - 59
      private int second;   // 0 - 59

/*Phương thức khởi tạo*/
       public Time( int h, int m, int s )
          {

/*Gọi hàm thiết lập giờ, phút, giây*/
         setTime( h, m, s );
         }

/*Định nghĩa hàm thiết lập giờ, phút, giây*/
      public void setTime( int h, int m, int s )
      {
         setHour( h );
         setMinute( m );
         setSecond( s );
      }

/* Định nghĩa hàm thiết lập giờ, nếu giờ > 23 thì thiết lập
lại là 0 */
      public void setHour( int h )
         {
         hour = ( ( h >= 0 && h < 24 ) ? h : 0 );
        }

/* Định nghĩa hàm thiết lập phút, nếu phút > 59 thì thiết lập
lại là 0 */
      public void setMinute( int m )
```

```
         {
         minute = ( ( m >= 0 && m < 60 ) ? m : 0 );
        }

/* Định nghĩa hàm thiết lập giây, nếu giây > 59 thì thiết lập
lại là 0 */
      public void setSecond( int s )
         {
         second = ( ( s >= 0 && s < 60 ) ? s : 0 );
        }

/*Hiển thị thời gian theo định dạng, nếu < 12 giờ thì hiển AM
phía sau và lớn hơn 12 giờ thì hiển thị thêm PM*/
      public String toString()
      {
         return ( ( hour == 12 || hour == 0 ) ? 12 : hour %
   12 ) +
                ":" + ( minute < 10 ? "0" : "" ) + minute +
                ":" + ( second < 10 ? "0" : "" ) + second +
                ( hour < 12 ? " AM" : " PM" );
      }

/*tang giay 1 don vi*/
      public void tick()
      {
         setSecond( second + 1 );

   /*Nếu giây tăng lên 60 thì hàm setSecond đã thiết lập giây
   = 0 và do đó cần tăng phút lên 1*/
         if ( second == 0 )
            incrementMinute();
      }

/*Tăng phút lên 1 đơn vị*/
      public void incrementMinute()
      {
         setMinute( minute + 1 );

   /*Nếu phút tăng lên 60 thì hàm setMinute đã thiết lập phút
   = 0 và do đó cần tăng giờ lên 1*/
         if ( minute == 0 )
            incrementHour();
      }

/*Tăng giờ lên 1 đơn vị, nếu giờ tăng lên 24 thì thiết lập về
0*/
      public void incrementHour()
      {
         setHour( hour + 1 );
```

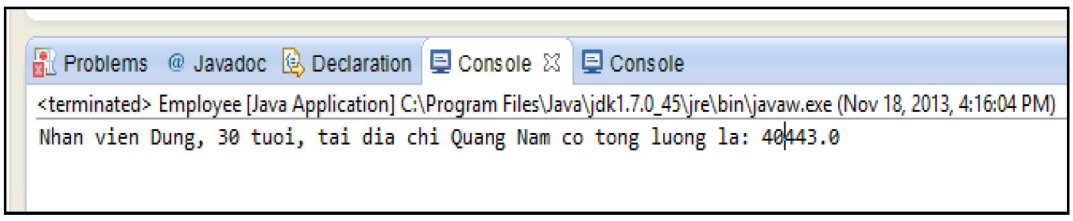
```
    }

    public static void main(String args[])
    {

/*Tạo đối tượng thời gian: 7 giờ, 0 phút, 0 giây*/
      Time A = new Time(7,0,0);
      System.out.println("Thoi gian da thiet lap =
"+A.toString());

/*Tạo vòng lặp vĩnh cửu*/
      while (true)
      {

/*Hiển thị thời gian hiện tại*/
        System.out.println(A.toString());

        try{

/*Dừng 1 giây = 1000 mili giây, hàm sleep phải gọi trong
khối try...catch*/
           Thread.sleep(1000);

/*Sau khi chờ 1 giây, tăng giây lên 1 đơn vị*/
          A.tick();
            }catch(Exception e)
               {
                }
      }
    }
}
```

6. Write an OOP program, creating a class Person with attributes *name, age, adress* and methods: *constructors, attributes information display*. Create an class

Employee inheriting the class Person. The class Employee has its own attributes *salary, rate* and methods *constructors, salary calculation and employee information display*.

```
/*Tạo lớp Person*/
class Person
{

/*Khai báo các thuộc tính cho lớp */
      private String name;
      private int age;
      private String address;

/*Phương thức khởi tạo*/
      public Person(String name, int age, String address)
      {

/*Khởi tạo giá trị cho các thuộc tính của đối tượng hiện đang
gọi phương thức này*/
            this.name=name;
            this.age =age;
            this.address =address;
      }

/*Hiển thị thông tin */
      public void display()
      {
      System.out.print("Nhan vien " + name+", "+age+" tuoi,
tai dia chi "+address);
      }
}

/*Tạo lớp Employee kế thừa lớp Person*/
public class Employee extends Person
{

/*Tạo thêm hai thuộc tính cho Employee*/
      private float salary;
      private float rate;

/*Định nghĩa hàm khởi tạo cho lớp Employee*/
  public Employee(String name, int age, String address, float
       salary, float rate)
      {

    /*Gọi hàm khởi tạo của lớp Person để khởi tạo thuộc tính*/
            super(name,age,address);

    /*Khởi tạo thuộc tính của Employee*/
```

```
            this.salary=salary;
            this.rate=rate;
      }

/*Định nghĩa hàm tính tổng lương*/
      public float totalSalary()
      {
            return salary*rate;
      }

/*Định nghĩa hàm hiển thị thông tin của Employee*/
      public void display()
      {

       /*Gọi hàm display của lớp Person để hiển thị name, age
và address*/
            super.display();

       /*Hiển thị thông tin lương*/
            System.out.print(" co tong luong la:
"+totalSalary());
      }
      public static void main(String[] args) {

       /*Tạo đối tượng của lớp Employee*/
       Employee A = new Employee("Dung", 30, "Quang
Nam",11050,3.66f);

       /*Hiển thị thông tin của A*/
      A.display();
      }
}
```



7. Write an OOP program, declaring the class Book to describe a book in the library. The class should have the following private attributes:

- title:java.lang.String
- author:Person (may use the class Person in the previous exercise)
- pages:int
- price:double

The Book class should also have the following public methods:

+ setTitle(java.lang.String):void
+ setAuthor(Person):void
+ setPages(int):void
+ setPrice(double):void
+ getTitle():java.lang.String
+ getAuthor():Person
+ getPages():int
+ getPrice():double

Students can add more constructors and methods to both of the classes Book and Person.

```
public class BookExercise
{
  public static void main(String args[])
    {
    Person author = new Person("HCPhap", 45, "VKU");
    Book OOP_Practice = new Book("OOP Practice with
                                  Java",author,200,50);
    OOP_Practice.details();
    }
}
class Book
{
  private String title;
  private Person author;
  private int pages;
  private double price;
  public Book(String title, Person author,int pages, double
               price)
    {
    this.title = title;
    this.author = author;
    this.pages = pages;
    this.price = price;
    }
  public void details()
    {
    System.out.println("Title : " + title);
    System.out.println("Author name : " + author.getName());
    System.out.println("Pages : " + pages);
    System.out.println("Price : " + price + "$");
    }
  public void setTitle(String title)
    {
    this.title = title;
    }
  public String getTitle()
    {
    return title;
```

```
    }
  public void setAuthor(Person person)
    {
    author = person;
    }
  public Person getAuthor()
    {
    return author;
    }
  public void setPages(int pages)
    {
    this.pages = pages;
    }
  public int getPages()
    {
    return pages;
    }
  public void setPrice(double price)
    {
    this.price = price;
    }
  public double getPrice()
    {
    return price;
    }
}
/*Tạo lớp Person*/
class Person
{

/*Khai báo các thuộc tính cho lớp */
     private String name;
     private int age;
     private String address;

/*Phương thức khởi tạo*/
     public Person(String name, int age, String address)
     {

/*Khởi tạo giá trị cho các thuộc tính của đối tượng hiện đang
gọi phương thức này*/
          this.name=name;
          this.age =age;
          this.address =address;
     }
     public String getName()
     {
     return name;
     }
```

```
/*Hiển thị thông tin */
     public void display()
     {
     System.out.print("Nhan vien " + name+", "+age+" tuoi, tai
dia chi "+address);
     }
}
```

```
Title : OOP Practice with Java
Author name : HCPhap
Pages : 200
Price : 50.0$
```

8. Write an OOP program, declaring a Hotel class that describes a hotel. The class should have the following private attributes:

- rooms:Room[]
- place:String

The Hotel class should have the following public methods:

+ Hotel(Room[],String)
+ setRooms(Room[]):void
+ setPlace(java.lang.String):void
+ getRooms():Room[]
+ getPlace():String
+ toString():java.lang.String

The Room class should have the following private attributes:

- beds:int
- tv:boolean
- telephone:boolean

The Room class should have the following public methods:

+ Room(int,boolean,boolean)
+ setBeds(int)
+ setTV(boolean)
+ setTelephone(boolean)
+ getBeds():int
+ getTV():boolean
+ getTelephone():boolean

Students can add more constructors and methods to both of the classes.

```
public class HotelExercise
{
 public static void main(String args[])
    {
     Room array[] = new Room[4];
     array[0] = new Room(2,false,true);
```

```
        array[1] = new Room(4,true,false);
        array[2] = new Room(4,false,false);
        array[3] = new Room(2,true,true);
        Hotel hotel = new Hotel(array,"Danang");
        System.out.println(hotel);
      }
}
class Room
{
    private int beds;
    private boolean tv;
    private boolean telephone;
    Room(int numOfBeds, boolean tvExists, boolean
          telephoneExists)
     {
     beds = numOfBeds;
     tv = tvExists;
     telephone = telephoneExists;
     }
     public String toString()
     {
      return "beds="+beds+" tv="+tv+"
                    telephone="+telephone+"\n";
     }
     public void setBeds(int num)
     {
      beds = num;
     }
     public int getBeds()
     {
     return beds;
     }
     public void setTv(boolean tvExists)
     {
     tv = tvExists;
     }
     public boolean getTv()
     {
     return tv;
     }
     public void setTelephone(boolean telephoneExists)
     {
      telephone = telephoneExists;
     }
     public boolean getTelephone()
     {
     return telephone;
     }
}
class Hotel
```

```
{
    private Room rooms[];
    private String place;
    Hotel(Room vec[])
     {
      rooms = vec;
     }
    Hotel(Room vec[], String place)
     {
      rooms = vec;
      this.place = place;
     }
    public void setRooms(Room vec[])
     {
      rooms = vec;
     }
    public void setPlave(String place)
     {
      this.place = place;
     }
    public Room[] getRooms()
     {
      return rooms;
     }
    public String getPlace()
     {
      return place;
     }
    public String toString()
     {
      StringBuffer sb = new StringBuffer();
      sb.append("place:");
      sb.append(place);
      sb.append("\n");
      for(int i=0;i<rooms.length; i++)
        {
      sb.append("room num.");
      sb.append(i+1);
      sb.append(":");
      sb.append(rooms[i]);
        }
      return sb.toString();
     }
}
```

```
place:Danang
room num.1:beds=2 tv=false telephone=true
room num.2:beds=4 tv=true telephone=false
room num.3:beds=4 tv=false telephone=false
room num.4:beds=2 tv=true telephone=true
```

II. Do it yourself

1. Using the Point2D class above, write an OOP program, creating a Rectangle2D class to describe a simple rectangle. The Rectangle2D class should have constructions, area, perimeter, display,...methods.
2. Using the class Point2D, write an OOP program, creating a Circle2D class that describes a simple circle. The Circle2D class should have the following private attributes and public methods:

   - radius : double
   - color: java.awt.Color
   - center: Point2D
   + Circle2D(Point2D, double, Color)
   + getRadius(): double
   + setRadius(double): double
   + getColor(): Color
   + setColor(Color): void
   + getCenter(): Point2D
   + setCenter(Point2D): void

   Students can add more constructors and methods.
3. Write an OOP program, declaring a class Mobile to describe a mobile telephone. The class should have the following private attributes and public methods:

   - ownerName : java.lang.String
   - color: java.awt.Color
   - number : String
   - model : java.lang.String
   + setOwnerName(java.lang.String):void
   + getOwnerName():java.lang.String
   + setColor(java.awt.Color):void
   + getColor():java.awt.Color
   + setNumber(String):void
   + getNumber():String
   + setModel(java.lang.String):void
   + getModel():java.lang.String
   + toString():java.lang.String

   Students can add more constructors and methods.
4. Write an OOP program, declaring a class Salary that describes a salary. The class should have the following private attributes and public methods:

   - sum:double
   - month:int
   - tax:double
   + setSum(double):void
   + setMonth(int):void
   + setTax(double):void

+ getSum():double
+ getMonth():int
+ getTax():double
+ toString():java.lang.String

Students can add more constructors and methods.

5. Write an OOP program, declaring a class Vacation that describes a vacation. The class should have the following private attributes and public methods:

- length:int
- date:java.util.Date
- place:String
+ setPlace(java.lang.String):void
+ setDate(java.util.Date):void
+ setLength(int):void
+ getPlace():String
+ getLength():int
+ getDate():java.util.Date
+ toString():java.lang.String

Students can add more constructors and methods.

6. Write an OOP program, creating classes as follows:

- a class Hotel class that describes a hotel. The class should have the following private attributes and public methods:

- rooms:Room[]
- name:java.lang.String
- place:Place
+ Hotel(Room[],java.lang.String, Place)
+ setRooms(Room[]):void
+ setName(java.lang.String):void
+ getRooms():Room[]
+ getName():java.lang.String
+ setPlace(Place):void
+ getPlace():Place
+ toString():java.lang.String

- The Place class should have the following private attributes and public methods as follows:

- cityName.java.lang.String
- countryName:java.lang.String
- currency:Currency
- cityHallTel:java.lang.String
+ Place(java.lang.String, java.lang.String, Currency, java.lang.String)
+ getCityName():java.lang.String
+ setCityName(java.lang.String):void
+ getCountryName():java.lang.String
+ setCountryName(java.lang.String):void

+ getCurrency():Currency
+ setCurrency(Currency):void

- The Currency class should have the following private attributes and pubic methods:

- name:java.lang.String
- currentDollarRate:double
+ Currency(java.lang.String, double)
+ setName(java.lang.String):void
+ getName():void
+ setCurrentDollarRate(double):void
+ getCurrentDollarRate():void
+ toString():java.lang.String

- The Room class should have the following private attributes and public methods:

- beds:int
- tv:boolean
- telephone:boolean

The Room class should have the following public methods:
+ Room(int,boolean,boolean)
+ setBeds(int)
+ setTV(boolean)
+ setTelephone(boolean)
+ getBeds():int
+ getTV():boolean
+ getTelephone():boolean

Students can add more constructors and methods to the classes.

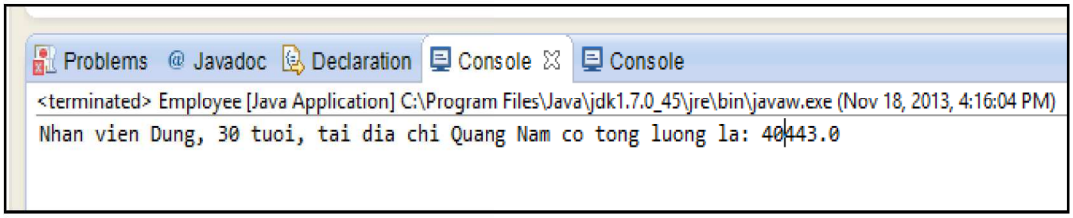
Chapter 6. Implementation of Inheritance and Polymorphism

I. Exercises with solutions

1. Write an OOP program, creating a class Person with attributes *name, age, adress* and methods: *constructors, attributes information display*. Create an class Employee inheriting the class Person. The class Employee has its own attributes *salary, rate* and methods *constructors, salary calculation and employee information display*.

```
/*Tạo lớp Person*/
class Person
{

/*Khai báo các thuộc tính cho lớp */
      private String name;
      private int age;
      private String address;

/*Phương thức khởi tạo*/
      public Person(String name, int age, String address)
      {

/*Khởi tạo giá trị cho các thuộc tính của đối tượng hiện đang
gọi phương thức này*/
            this.name=name;
            this.age =age;
            this.address =address;
      }

/*Hiển thị thông tin */
      public void display()
      {
      System.out.print("Nhan vien " + name+", "+age+" tuoi,
tai dia chi "+address);
      }
}

/*Tạo lớp Employee kế thừa lớp Person*/
public class Employee extends Person
{

/*Tạo thêm hai thuộc tính cho Employee*/
      private float salary;
      private float rate;

/*Định nghĩa hàm khởi tạo cho lớp Employee*/
  public Employee(String name, int age, String address, float
       salary, float rate)
```

```
      {

    /*Gọi hàm khởi tạo của lớp Person để khởi tạo thuộc tính*/
           super(name,age,address);

    /*Khởi tạo thuộc tính của Employee*/
          this.salary=salary;
          this.rate=rate;
      }

/*Định nghĩa hàm tính tổng lương*/
      public float totalSalary()
      {
          return salary*rate;
      }

/*Định nghĩa hàm hiển thị thông tin của Employee*/
      public void display()
      {

       /*Gọi hàm display của lớp Person để hiển thị name, age
và address*/
          super.display();

       /*Hiển thị thông tin lương*/
           System.out.print(" co tong luong la:
"+totalSalary());
      }
      public static void main(String[] args) {

       /*Tạo đối tượng của lớp Employee*/
       Employee A = new Employee("Dung", 30, "Quang
Nam",11050,3.66f);

       /*Hiển thị thông tin của A*/
      A.display();
      }
}
```



2. Write an OOP program, creating the following classes:

+ an abstract class Shape

+ a class Rectangle, that extends Shape, describes a simple rectangle.

+ a class Circle, that extends Shape, describes a simple circle.

The class Shape class declares an abstract method of area. The classes Rectangle and Circle implement the abstract method of area inherited from class Shape.

```
abstract class Shape
{
  public abstract double area();
  public String toString()
    {
     return "The area is " + area();
    }
}
class Rectangle extends Shape
{
   private double width, height;
   public Rectangle(double wVal, double hVal)
    {
     width = wVal;
     height = hVal;
    }
   public double area()
    {
    return width*height;
    }
}
class Circle extends Shape
{
   private double radius;
   public Circle(double rad)
    {
    radius = rad;
    }
   public double area()
    {
    return Math.PI * radius * radius;
    }
}
public class AbstractShape
{
   public static void main(String args[])
    {
    Shape vec[] = {new Circle(5), new Rectangle(4,5), new
Circle(4), new Rectangle(7,8)};
    for(int index = 0; index < vec.length; index ++)
```

```
        {
        System.out.println(vec[index]);
        }
      }
}
```

3. Write an OOP program, creating the following classes:

+ an abstract class Employee

+ a Manager class, that extends Employee, describes a manager.

+ a Clerk class, that extends Employee, describes a clerk.

The class Employee class declares an abstract method of salary. The classes Manager and Clerk implement the abstract method of salary inherited from class Employee.

```
abstract class Employee
{
   private String name;
   private double age;
   private double hourRate;
   public Employee(String name, double age, double hourRate)
      {
      this.name = name;
      this.age = age;
      this.hourRate = hourRate;
      }

   public abstract double salary(double hours);

   public String getName()
   {
     return name;
   }
   public double getHourRate()
   {
     return hourRate;
   }
   public String toString()
    {
    return "name= " + name + " age=" + age + " hourRate=" +
           hourRate;
    }
}
class Manager extends Employee
{
  public  Manager(String name, double age, double hourRate)
```

```
    {
     super(name, age, hourRate);
    }
  public double salary(double hours)
    {
     return hours * this.getHourRate() * 1.2;
    }
}
class Clerk extends Employee
{
   public Clerk(String name, double age, double hourRate)
    {
     super(name, age, hourRate);
    }
   public double salary(double hours)
    {
    return hours * this.getHourRate() * 0.8;
    }
}
public class AbstractEmployee
{
   public static void main(String args[])
     {
     Employee vec[] = {new Manager("HCPhap", 45, 104), new
Clerk("Jonh Le", 26, 110), new Manager("David Huynh", 42,
120), new Manager("Marry Nguyen", 34, 120)};
     double hours[] = {140,150,130,180};
     double total = 0;
     for(int index = 0; index < vec.length; index ++)
      {
      total += vec[index].salary(hours[index]);
      System.out.println(vec[index]);
      }
     System.out.println("The total payment of the employees is
" + total);
    }
}
```

```
name= HCPhap age=45.0 hourRate=104.0
name= Jonh Le age=26.0 hourRate=110.0
name= David Huynh age=42.0 hourRate=120.0
name= Marry Nguyen age=34.0 hourRate=120.0
The total payment of the employees is 75312.0
```

4. Write an OOP program, creating 10 objects of the following classes: Rectangle, Circle and EquilateralTriangle. Each of these classes should extend the abstract class

Shape in the previous exercise. Students should declare the three classes and write code to sort 10 created objects according to their area.

```
import java.util.Arrays;
abstract class Shape implements Comparable
{
  abstract double area();

  /*implement method compareTo of class Comparable
   * in order to compare two objects according to their area
   */
  public int compareTo(Object other)
    {
    return (int)(this.area()-((Shape)other).area());
    }
  public String toString()
    {
    return "area="+area();
    }
}
class Circle extends Shape
{
    double radius;
    Circle(double radiusVal)
      {
      radius = radiusVal;
      }
    double area()
     {
      return Math.PI * Math.pow(radius,2);
     }
    public String toString()
     {
      return "Circle:"+super.toString();
     }
}
class Rectangle extends Shape
{
    double width, height;
    Rectangle(double widthVal, double heightVal)
    {
     width = widthVal;
     height = heightVal;
    }
   double area()
    {
    return width*height;
    }
   public String toString()
```

```
    {
    return "Rectangle:"+super.toString();
    }
}
class EquilateralTriangle extends Shape
{
    double length;
    EquilateralTriangle(double length)
     {
     this.length = length;
     }
    double area()
     {
     return 0.5*length*Math.sin((2*60*Math.PI/360));
     }
    public String toString()
     {
     return "EquilateralTriangle:"+super.toString();
     }
}
public class ShapeSort
{
   public static void main(String args[])
     {
     Shape vec[] = new Shape[10];

     //automatically initialize values for objects
     for(int index=0; index<vec.length; index++)
      {
      switch((int)(4*Math.random()))
        {
        case 1:
         vec[index] = new Rectangle(1000*Math.random(),
1000*Math.random());
         break;
        case 2:
          vec[index] = new Circle(1000*Math.random());
         break;
        default:
         vec[index] = new
EquilateralTriangle(1000*Math.random());
        }
    }

     Arrays.sort(vec);

        for(int index=0; index<vec.length; index++)
        {
          System.out.println(vec[index]);
        }
```

```
    }
}
```

```
Rectangle:area=158685.0192807511
Circle:area=292793.5868513119
Rectangle:area=569295.2494713358
Circle:area=601084.1846221576
Circle:area=770995.0773476631
Circle:area=928398.7038432377
```

5. Write an OOP program, creating an interface Vehicle declaring abstract methods. Creating a class Motobike and a class Car implementing the interface Vehicle by overriding all abstract inherited methods.

```
interface Vehicle
{
      // all are the abstract methods.
      public void changeGear(int a);
      public void speedUp(int a);
      public void brake(int a);
}

class MotoBike implements Vehicle{

      private String brand;
      private int speed;
      private int gear;
      private int wheels;

      public MotoBike(String brand,int speed, int gear, int
wheels)
      {
       this.brand=brand;
       this.speed=speed;
       this.gear=gear;
       this.wheels=wheels;
      }
      // to change gear, overide
      public void changeGear(int newGear)
      {
       gear = newGear;
      }

      // to increase speed, Override
      public void speedUp(int increment)
      {
       speed = speed + increment;
```

```
    }

    // to decrease speed, Override
    public void brake(int decrement)
    {
     speed = speed - decrement;
    }

    public void printStates() {
     System.out.println("Brand: "+brand+", wheels:
"+wheels+", speed: " + speed + ", gear: " + gear);
    }
}

class Car implements Vehicle {

    private String brand;
    private int speed;
    private int gear;
    private int wheels;

    public Car(String brand,int speed, int gear, int
wheels)
    {
        this.brand=brand;
        this.speed=speed;
        this.gear=gear;
        this.wheels=wheels;
    }

    // to change gear, Override
    public void changeGear(int newGear)
    {
        gear = newGear;
    }
    // to increase speed, Override
    public void speedUp(int increment)
    {
        speed = speed + increment;
    }

    // to decrease speed, Override
    public void brake(int decrement)
    {
        speed = speed - decrement;
    }

    public void printStates() {
        System.out.println("Brand: "+brand+", wheels:
"+wheels+", speed: " + speed + ", gear: " + gear);
```

```
    }

}
class VehicleInterface {

    public static void main (String[] args) {

        // creating an inatance of Bicycle
        // doing some operations
        MotoBike bike = new MotoBike("Honda",100,4,2);
        bike.changeGear(2);
        bike.speedUp(3);
        bike.brake(1);

        System.out.println("Motobike:");
        bike.printStates();

        // creating instance of the bike.
        Car car = new Car("Mitsubishi",300,6,4);
        car.changeGear(1);
        car.speedUp(4);
        car.brake(3);

        System.out.println("Car:");
        car.printStates();
    }
}
```

```
Motobike:
Brand: Honda, wheels: 2, speed: 102, gear: 2
Car:
Brand: Mitsubishi, wheels: 4, speed: 301, gear: 1
```

6. Write an OOP program, creating an interface Message declaring abstract methods. Creating a class TextMessage and a class BinaryMessage implementing the interface Message by overriding all abstract inherited methods.

```
interface Message
{
    public String getHeader();
    public void    setHeader(String str);

    public String getBody();
```

```
      public void    setBody(String str);

      public String getType();
      public void    setType(String str);
      public void printMessage();
}
class TextMessage implements Message
{
      String header;
      String body;
      String type;
      public TextMessage(String header, String body)
      {
       this.body=body;
       this.header =header;
       this.type="text";
      }
      public String getHeader()
      {
       return header;
      }
      public void    setHeader(String str)
      {
       header=str;
      }

      public String getBody()
      {
       return body;
      }
      public void    setBody(String str)
      {
      body=str;
      }

      public String getType()
      {
       return type;
      }
      public void    setType(String str)
      {
       type=str;
      }
      public void printMessage()
      {
      System.out.println("Type: "+type+"\nheader:
"+header+"\nbody: "+body);
      }
      }
class BinaryMessage implements Message
```

```
{
      String header;
      String body;
      String type;
      public BinaryMessage(String header, String body)
      {
       this.body=body;
       this.header =header;
       this.type="binary";
      }
      public String getHeader()
      {
       return header;
      }
      public void    setHeader(String str)
      {
       header=str;
      }

      public String getBody()
      {
       return body;
      }
      public void    setBody(String str)
      {
      body=str;
      }

      public String getType()
      {
       return type;
      }
      public void    setType(String str)
      {
       type=str;
      }
      public void printMessage()
      {
      System.out.println("Type: "+type+"\nheader:
"+header+"\nbody: "+body);
      }
}


class MessageInterface {

      public static void main (String[] args) {
            TextMessage msg1 = new
TextMessage("time2021:oganizerVKU:createrHCPhap","This is an
text message created by HCPhap-VKU");
```

```
            BinaryMessage msg2 = new
BinaryMessage("time2021:oganizerVKU:createrHCPhap","10101010
10101010111110101");

            msg1.printMessage();
            msg2.printMessage();
      }
}
```

```
Type: text
header: time2021:oganizerVKU:createrHCPhap
body: This is an text message created by HCPhap-VKU
Type: binary
header: time2021:oganizerVKU:createrHCPhap
body: 10101010101010111110101
```

7. Write an OOP program of students management, including basic functionalities as follows:

- Input a list of students,
- Display information of the list of students
- Sort the students list according to their average points,
- Searh a student by name.

```
import java.util.Scanner;
import java.util.Vector;
import java.util.Enumeration;
import java.util.Arrays;

public class StudentManagement
{

/*dung de chua danh sach sinh vien*/
 Vector list = new Vector();

 public StudentManagement()
 {
  while(true)
  {

/*Hien thi menu chuong trinh*/
   System.out.println("*-CHUONG TRINH QUAN LY SINH VIEN-*");
   System.out.println("*-Chuc nang chinh chuong trinh-*");
   System.out.println("  1. Nhap danh sach sinh vien   ");
   System.out.println("  2. Xem danh sach sinh vien   ");
   System.out.println("  3. Sap xep sach sinh vien tang dan
diem trung binh   ");
```

```
   System.out.println("  4. Tim sinh vien theo ten   ");
   System.out.println("  5. Thoat   ");
   System.out.println("  -------------");

 /*Nhập một số từ bàn phím*/
   int num;
   Scanner keyboard = new Scanner(System.in);
   System.out.print("  Nhap mot so de chon chuc nang: ");
   num = keyboard.nextInt();

/*Kiểm tra và gọi chức năng tương ứng*/
   switch(num)
   {
   case 1:
        this.input();
        break;
   case 2:
        this.view();
        break;
   case 3:
        sort();
        break;
   case 4:
        search();
        break;
   case 5:
        System.out.print("----  Chuong trinh ket thuc----- ");
        return;
        }
  }
 }

/*Nhap danh sach sinh vien*/
   public void input()
   {

   /*Nhập số lượng sinh viên cho danh sách*/
      int num;
      Scanner keyboard = new Scanner(System.in);
      System.out.print("  Nhap so luong sinh vien: ");
      num = keyboard.nextInt();

   /*Nhập thông tin cho mỗi sinh viên*/
      for (int i=1;i<=num;i++)
      {
      System.out.println("  Nhap thong tin cho sinh vien thu:
"+i);
      System.out.print("  ID: ");
      int id = Integer.parseInt(keyboard.next());
```

```
      keyboard.nextLine();//xoa bo dem
      System.out.print("  Ten: ");
      String name = keyboard.nextLine();

      System.out.print("  Diem trung binh: ");
      float aver = keyboard.nextFloat();

  /*Sau khi nhập thông tin, tạo đối tượng sinh viên*/
      Student st = new Student(id,name,aver);

  /*Lưu đối tượng sinh viên vào danh sách*/
         list.add(st);
     }
     System.out.println("\n--------\n");
    }

/*Xem danh sach sinh vien*/
    public void view()
    {

    /*Hiển thị danh sách sinh viên*/
     System.out.println("  Thong tin danh sach sinh vien");

    /*Lấy sinh viên từ danh sách (vector) và lưu trữ ở vEnum*/
     Enumeration vEnum = list.elements();

    /*Duyệt từng phần tử của vEnum*/
     int i=1;

    /*Chưa hết phần tử*/
     while(vEnum.hasMoreElements())
      {

     /*Lấy phần tử từ vEnum ép lại kiểu Student*/
      Student sts = (Student)vEnum.nextElement();

     /*Hiển thị thông tin sinh viên*/
      System.out.println("    "+i+". ID="+sts.getId()+",
Ten="+sts.getName()+", Diem trung binh="+sts.getAver());
         i++;
      }
       System.out.println("\n--------\n");
    }

/*sap xep danh sach theo chieu tang dan cua diem trung binh
su dung ham sort cuar lop Arrays*/
    public void sort()
    {
```

```
   /*Đổ dữ liệu từ Vector vào mảng để gọi hàm sort sắp xếp
mảng*/
     Student[] sts = new Student[list.size()];
     int index=0;

     Enumeration vEnum = list.elements();
     while(vEnum.hasMoreElements())
      {
       sts[index] = (Student)vEnum.nextElement();
       index++;
      }

    /*Sắp xếp mảng*/
      Arrays.sort(sts);

       System.out.println("\n--Danh sach sinh vien sau khi sap
xep--");
     for(index=0; index < sts.length; index++)
        {

    /*Hiển thị thông tin sinh viên sau khi sắp xếp*/
       System.out.println("    "+(index+1)+".
ID="+sts[index].getId()+", Ten="+sts[index].getName()+", Diem
trung binh="+sts[index].getAver());
        }
       System.out.println("\n--------\n");

        }

/*Tìm kiếm sinh viên theo tên*/
   public void search()
   {

    /*Nhập tên sinh viên cần tìm kiếm*/
      Scanner keyboard = new Scanner(System.in);
      System.out.print(" Nhap ten sinh vien can tim: ");
      String name = keyboard.nextLine();

   /*Duyệt từng phần tử của mảng để so sánh tên tìm kiếm*/
      Enumeration vEnum = list.elements();

       System.out.println("\n--Thong tin tim kiem duoc--");
      while(vEnum.hasMoreElements())
      {
      Student sts = (Student)vEnum.nextElement();

       /*Nếu tên sinh viên chứa chuỗi nhập vào thì hiển thị
    thông tin đối     tượng sinh viên*/
```

```
      if (sts.getName().indexOf(name)!=-1)
        System.out.println("ID="+sts.getId()+",
Ten="+sts.getName()+", Diem trung binh="+sts.getAver());
      }
        System.out.println("\n--------\n");

   }

 public static void main(String[] args)
   {
  /*Khởi tạo chương trình*/
    new StudentManagement();
   }
}

/*Xây dựng lớp Student hiện thực interface Comparable, định
nghĩa cụ thể hàm compareTo để phục vụ sắp xếp*/



class Student implements Comparable
{
      private int id;
      private String name;
      private float aver;

/*Hàm khởi tạo mặc định*/
      public Student()
      {
            name = new String("");
            id = 0;
            aver=0;
      }

/*Hàm khởi tạo 3 đối số*/
      public Student(int i, String n, float a)
      {
            id = i;
            name = n;
            aver=a;
      }

/*Hàm lấy giá trị name*/
      public String getName()
      {
            return name;
      }

/*Hàm lấy giá trị id*/
```

```
	public int getId()
	{
		return id;
	}

/*Hàm lấy giá trị Aver*/
	public float getAver()
	{
		return aver;
	}

/*Hàm so sánh 2 đối tượng Student phục vụ sắp xếp*/
	public int compareTo(Object other)
	   {
	    Student otherRect = (Student)other;
	    return (int)(this.aver-otherRect.aver);
	   }
}
```

```
Problems  @ Javadoc  Declaration  Console  Console
<terminated> StudentManagement (1) [Java Application] C:\Program Files\Java
*-CHUONG TRINH QUAN LY SINH VIEN-*
*-Chuc nang chinh chuong trinh-*
  1. Nhap danh sach sinh vien
  2. Xem danh sach sinh vien
  3. Sap xep sach sinh vien tang dan diem trung binh
  4. Tim sinh vien theo ten
  5. Thoat
  -------------
  Nhap mot so de chon chuc nang: 1
  Nhap so luong sinh vien: 3
  Nhap thong tin cho sinh vien thu: 1
  ID: 1
  Ten: hcphap
  Diem trung binh: 8
  Nhap thong tin cho sinh vien thu: 2
  ID: 2
  Ten: ddtho
  Diem trung binh: 9
  Nhap thong tin cho sinh vien thu: 3
  ID: 3
  Ten: lmthai
  Diem trung binh: 9.5

 --------
```

II. Do it yourself

1. Write an OOP program, creating the following classes:

+ an abstract class Shape, declaring abstract methods: area, perimeter, draw.
+ a class Square, that extends Shape, describes a simple square.
+ a class Circle, that extends Shape, describes a simple circle.
+ a class EquilateralTriangle, that extends Shape, describes an equilateral triangle.

2. Write an OOP program, creating an interface Animal declaring abstract methods. Creating a class Dog and a class Cat implementing the interface Animal by overriding all abstract inherited methods.
3. Write an OOP program, reusing the classes TextMessage and BinaryMessage above, creating a class MessageProducer and a class MessageConsumer. The class MessageProducer includes methods *create, send* and other methods. The class MessageConsumer includes methods *receive, show* and other methods.
4. Run all exercises with solutions and work yourself all exercises in **Chapter 2 of the Book "Bài tập lập trình Java Cơ bản –** Có lời giải**"**

Chapter 7. Exception Handling

I. Exercises with solutions

1. Run all exercises with solutions in **Chapter 3. Xử lý biệt lệ (Exception Handling) of the Book "Bài tập lập trình Java Cơ bản –** Có lời giải**"** by Huynh Cong Phap.
2. Read more about Regular Expressions.

II. Do it yourself

1. Solve all exercises yourself in the section of exercises without solution in **Chapter 3. Xử lý biệt lệ (Exception Handling) of the Book "Bài tập lập trình Java Cơ bản –** Có lời giải**"** by Huynh Cong Phap.

Chapter 8. GUI programming with Java

I. Exercises with solutions

1. Run all exercises with solutions in **Chapter 4. Lập trình giao diện với AWT và Swing (GUI Programming) of the Book "Bài tập lập trình Java Cơ bản – Có lời giải"** by Huynh Cong Phap.

II. Do it yourself

2. Solve all exercises yourself in the section of exercises without solution in **Chapter 4. Lập trình giao diện với AWT và Swing (GUI Programming) of the Book "Bài tập lập trình Java Cơ bản – Có lời giải"** by Huynh Cong Phap.

Chapter 9. Java Database Connectivity

I. Exercises with solutions

1. Run all exercises with solutions in **Chapter 8. Lập trình truy xuất cơ sở dữ liệu (Java Database Connectivity) of the Book "Bài tập lập trình Java Cơ bản –** Có lời giải**"** by Huynh Cong Phap.

II. Do it yourself

2. Solve all exercises yourself in the section of exercises without solution in **Chapter 8. Lập trình truy xuất cơ sở dữ liệu (Java Database Connectivity) of the Book "Bài tập lập trình Java Cơ bản –** Có lời giải**"** by Huynh Cong Phap.