Course Code : BCSL-043

Course Title : Java Programming Lab

Assignment Number : BCA(4)/L-043/Assignment/16-17

Maximum Marks : 50 Weightage : 25%

Last Dates for Submission : 15 th October, 2016 (For July 2016 Session)

15th April, 2017 (For January 2017 Session) This assignment has three questions. Answer all the questions. These questions carry 40 marks. Rest 10 marks are for viva voce. You are advised to give proper comments and do proper alignments while writing java program. Please go through the guidelines regarding the assignments given in the programme guide for the format of presentation.

1. Write and execute java program which create a shape class and derive circle and rectangle classes from shape class. All the classes in this program should have proper constructors and methods to display details of different shapes. Also use appropriate access specifiers in your program. (10 Marks)

```
nhgrif says:
 * Phrancis ready for inheritance/polymorphism?
 * Given the following abstract class:
 *
 *  public abstract class Shape {
 *      public abstract double area();
 *      public abstract double perimeter();
 *  }
 *
 * Implement a Circle, Triangle, and Rectangle class which extend the class Shape.
 * Ex: public class Circle extends Shape ... etc
 */

public abstract class Shape {
    public abstract double area();
    public abstract double perimeter();
}
public class Rectangle extends Shape {
    private final double width, length; //sides

    public Rectangle() {
        this(1,1);
    }
    public Rectangle(double width, double length) {
        this.width = width;
        this.length = length;
    }

    @Override
    public double area() {
        // A = w * l
        return width * length;
    }

    @Override
    public double perimeter() {
        // P = 2(w + l)
        return 2 * (width + length);
```

```
    }

}
public class Circle extends Shape {
    private final double radius;
    final double pi = Math.PI;

    public Circle() {
        this(1);
    }
    public Circle(double radius) {
        this.radius = radius;
    }

    @Override
    public double area() {
        // A = π r^2
        return pi * Math.pow(radius, 2);
    }

    public double perimeter() {
        // P = 2πr
        return 2 * pi * radius;
    }
}
public class Triangle extends Shape {
    private final double a, b, c; // sides

    public Triangle() {
        this(1,1,1);
    }
    public Triangle(double a, double b, double c) {
        this.a = a;
        this.b = b;
        this.c = c;
    }

    @Override
    public double area() {
        // Heron's formula:
        // A = SquareRoot(s * (s - a) * (s - b) * (s - c))
        // where s = (a + b + c) / 2, or 1/2 of the perimeter of the triangle
        double s = (a + b + c) / 2;
        return Math.sqrt(s * (s - a) * (s - b) * (s - c));
    }

    @Override
    public double perimeter() {
        // P = a + b + c
        return a + b + c;
    }
}
public class TestShape {
    public static void main(String[] args) {

        // Rectangle test
        double width = 5, length = 7;
        Shape rectangle = new Rectangle(width, length);
        System.out.println("Rectangle width: " + width + " and length: " + length
                + "\nResulting area: " + rectangle.area()
                + "\nResulting perimeter: " + rectangle.perimeter() + "\n");

        // Circle test
        double radius = 5;
        Shape circle = new Circle(radius);
        System.out.println("Circle radius: " + radius
            + "\nResulting Area: " + circle.area()
            + "\nResulting Perimeter: " + circle.perimeter() + "\n");

        // Triangle test
        double a = 5, b = 3, c = 4;
        Shape triangle = new Triangle(a,b,c);
        System.out.println("Triangle sides lengths: " + a + ", " + b + ", " + c
                + "\nResulting Area: " + triangle.area()
```

```
                + "\nResulting Perimeter: " + triangle.perimeter() + "\n");
    }
}
```

2. Write a program in java for exception handling for operating a Stack data structure. Throw appropriate exceptions in different cases (such as stack is full and attempt is made to push the data etc.) (15 Marks)

```
import java.io.FileNotFoundException;

import java.io.IOException;



public class ExceptionHandling {



        public static void main(String[] args) throws FileNotFoundException,
IOException {

                try{

                        testException(-5);

                        testException(-10);

                }catch(FileNotFoundException e){

                        e.printStackTrace();

                }catch(IOException e){

                        e.printStackTrace();

                }finally{

                        System.out.println("Releasing resources");

                }

                testException(15);

        }



        public static void testException(int i) throws FileNotFoundException,
IOException{
```

```
                if(i < 0){

                        FileNotFoundException myException = new
FileNotFoundException("Negative Integer "+i);

                        throw myException;

                }else if(i > 10){

                        throw new IOException("Only supported for index 0 to
10");

                }



        }



}
```

3. Write a program in java to create an applet which draws a circle and display current date and time inside the circle.

```
import java.awt.*;
import java.applet.*;

import java.applet.*;
import java.awt.*;
import java.util.*;

public class ClockApplet extends Applet implements Runnable{
   Thread t,t1;
   public void start(){
      t = new Thread(this);
      t.start();
   }
   public void run(){
      t1 = Thread.currentThread();
      while(t1 == t){
         repaint();
         try{
            t1.sleep(1000);
         }
```

```
         catch(InterruptedException e){}
      }
   }
   public void paint(Graphics g){
      Calendar cal = new GregorianCalendar();
      String hour = String.valueOf(cal.get(Calendar.HOUR));
      String minute = String.valueOf(cal.get(Calendar.MINUTE));
      String second = String.valueOf(cal.get(Calendar.SECOND));
      g.drawString(hour + ":" + minute + ":" + second, 20, 30);
   }
}
```