Week-5

Name – Randrita Sarkar [11500219058]

IT PCC-CS593 L - OBJECT ORIENTED PROGRAMMING LAB

1.Create a class for employee with attributes name, age and address. Create a class manager andworker,which will inherit the employee class. The manager class has extra attributes – dep artment and salary.The worker class has attributes – no. of days worked, daily wages and tota l salary. Add necessary number
of functions to calculate total salary of worker and display output of all classes.

```
package com.randrita.week5;
import java.util.Scanner;

public class Office {

    public static void main(String[] args) {

        Worker obj1 = new Worker();
        String Name=obj1.Name("Rimi");
        String Address=obj1.Address("32/B,JKM Road,Kolkata-
700037");
        int age= obj1.Age(21);
        int noDaysWorked= obj1.nDaysWorked(20);
        int dailyWage = obj1.DailyWage(1000);
        int totalSalary =
obj1.totalSalary(noDaysWorked,dailyWage);

        System.out.println("Employee Details(Output Employee
Class):\nName: " +Name +"\nAddress: "+Address +"\nAge: "+age );
        System.out.println("-------------------------------------
-------------------------------------");
        System.out.println("Manager Details(Output Manager
Class):\nDepartment: " + obj1.Department("IT") +"\nSalary: "+
obj1.Salary(100000));
        System.out.println("-------------------------------------
-------------------------------------");
        System.out.println("Worker Details(Output Worker
Class):\ntotalSalary: " + totalSalary );

    }
}

class Employee{

    public String Name(String name) {
        return name;
    }

    public String Address(String address) {
        return address;
```

```
    }

    public int Age(int age){
        return age;
    }
}

class Manager extends Employee{
    public String Department(String department) {
        return department;
    }
    public int Salary(int salary){
        return salary;
    }

}

class Worker extends Manager{

    public int nDaysWorked(int n_days_worked){
        return n_days_worked;
    }

    public int totalSalary(int a,int b){
        int total_salary= a*b;
        return total_salary;
    }

    public int DailyWage(int daily_wage){
        return daily_wage;
    }

}
```

Output:

```
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains
Employee Details(Output Employee Class):
Name: Rimi
Address: 32/B,JKM Road,Kolkata-700037
Age: 21
---------------------------------------------------------------------------
Manager Details(Output Manager Class):
Department: IT
Salary: 100000
---------------------------------------------------------------------------
Worker Details(Output Worker Class):
totalSalary: 20000

Process finished with exit code 0
```

2. Write a superclass called `Shape` (as shown in the class diagram), which contains:

- Two instance variables `color` (`String`) and `filled` (`boolean`).
- Two constructors: a no-arg (no-argument) constructor that initializes the `color` to "green" and `filled` to `true`, and a constructor that initializes the `color` and `filled` to the given values.
- Getter and setter for all the instance variables. By convention, the getter for a `boolean` variable `xxx` is called `isXXX()` (instead of `getXxx()` for all the other types).
- A `toString()` method that returns "`A Shape with color of xxx and filled/Not filled`".

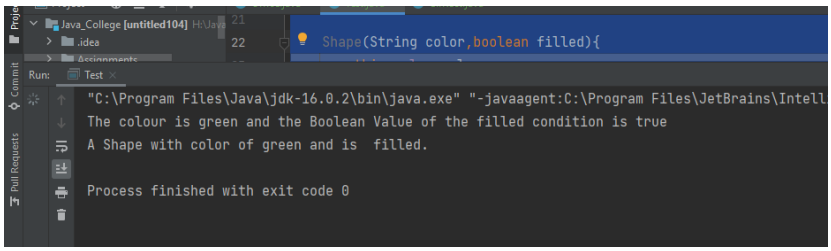
Write a test program to test all the methods defined in `Shape`.

Part 1 :

```
package com.randrita.week5;

public class Test {
    public static void main(String[] args) {
        Shape obj1 = new Shape();
        String s =obj1.getColor();
        System.out.printf("The colour is %s and the Boolean Value
of the filled condition is %b \n",obj1.getColor() ,
obj1.isFilled());
        System.out.println(obj1.toString());
    }
}

class Shape{
    String color;
    boolean filled;

    Shape(){
        color = "green";
        filled = true;

    }

    Shape(String color,boolean filled){
        this.color=color;
        this.filled=filled;
    }

    public void setColor(String colorSet) {
        this.color = colorSet;
    }

    public String getColor() {
        return color;
    }
```

```
    public boolean isFilled()
    {
        if (filled == true)
        {
            return true;
        }
        else
        {
            return false;
        }
    }

    public void setFilled(boolean filledSet)
    {
        this.filled = filledSet;
    }

    public String toString()
    {
        String isNot = "";
        if(isFilled() == false)
        {
            isNot = "not ";
        }
        return "A Shape with color of " + color + " and is " +
isNot + " filled. ";
    }
}
```

output:



Part:2

Write two subclasses of `Shape` called `Circle` and `Rectangle`, as shown in the class diagram.

The `Circle` class contains:

- An instance variable `radius` (`double`).
- Three constructors as shown. The no-arg constructor initializes the radius to `1.0`.
- Getter and setter for the instance variable `radius`.
- Methods `getArea()` and `getPerimeter()`.

- Override the `toString()` method inherited, to return "`A Circle with radius=xxx, which is a subclass of yyy`", where `yyy` is the output of the `toString()` method from the superclass.

The `Rectangle` class contains:

- Two instance variables `width` (`double`) and `length` (`double`).
- Three constructors as shown. The no-arg constructor initializes the `width` and `length` to `1.0`.
- Getter and setter for all the instance variables.
- Methods `getArea()` and `getPerimeter()`.
- Override the `toString()` method inherited, to return "`A Rectangle with width=xxx and length=zzz, which is a subclass of yyy`", where `yyy` is the output of the `toString()` method from the superclass.

Write a class called `Square`, as a subclass of `Rectangle`. Convince yourself that `Square` can be modeled as a subclass of `Rectangle`. `Square` has no instance variable, but inherits the instance variables width and length from its superclass Rectangle.

- Provide the appropriate constructors (as shown in the class diagram). Hint:
- `public Square(double side) {`
- `   super(side, side);  // Call superclass Rectangle(double, double)`
  `}`

- Override the `toString()` method to return "`A Square with side=xxx, which is a subclass of yyy`", where `yyy` is the output of the `toString()` method from the superclass.
- Do you need to override the `getArea()` and `getPerimeter()`? Try them out.
- Override the `setLength()` and `setWidth()` to change both the `width` and `length`, so as to maintain the square geometry.

```
• package com.randrita.week5;

  public class CirRect {
      public static void main(String[] args) {
          Shapee s = new Shapee();
          Shapee s1 = new Shapee("Orange",true);
          s1.toString();
          Circle c = new Circle();
          c.setRadius(5);
          System.out.println("Perimeter of circle is:" +
  c.getPerimeter(c.getRadius(5)));
          System.out.println("Area of circle is:" +
  c.getArea(c.getRadius(5)));

          Rectangle r = new Rectangle();
          r.setLength(5);
          r.setWidth(10);
          System.out.println("Perimeter of rectangle is:" +
  r.getPerimeter(5,5));
          System.out.println("Area of rectangle is:" +
```

```
r.getArea(5,5));

        Square sq = new Square(5);
        System.out.println("Perimeter of square is:" +
sq.getPerimeter());
        System.out.println("Area of square is:" +
sq.getPerimeter());


    }
}

class Shapee {
    String color;
    boolean filled;

    Shapee() {
        color = "green";
        filled = true;
    }

    Shapee(String color, boolean filled) {
        this.color = color;
        this.filled = filled;
    }

    public void setColor(String colorSet) {
        this.color = colorSet;
    }

    public String getColor() {
        return color;
    }

    public boolean isFilled() {
        if (filled == true) {
            return true;
        } else {
            return false;
        }
    }

    public void setFilled(boolean filledSet) {
        this.filled = filledSet;
    }

    public String toString() {
        String isNot = "";
        if (isFilled() == false) {
            isNot = "not ";
        }
        return "A Shape with color of " + color + " and is "
+ isNot + " filled. ";
```

```
    }
}

class Circle extends Shapee{
    double radius;
    Circle(){
        this.radius=1.0;
    }
    Circle(double radius,String color,boolean filled){
        this.radius=radius;
        this.color=color;
        this.filled=filled;
    }
    public void setRadius(double radius){
        this.radius=radius;
    }

    public double getRadius(double radius){
        return this.radius;
    }

    public double getArea(double radius){
        return (Math.PI*this.radius*this.radius);
    }

    public double getPerimeter(double radius){
        return (Math.PI*this.radius*2);
    }

    public String toString(){
        String s = "A Circle with radius" + radius + "which
is a subclass of" + super.toString();
        return s;
    }


}

 class Rectangle extends Shapee{
    double width;
    double length;

     Rectangle()
     {
         super();
         width = 1.0;
         length = 1.0;
     }

     Rectangle(double width, double length)
     {
         super();
         this.width = width;
```

```
            this.length = length;
        }


        Rectangle(double width,double length,String
color,boolean filled){
            super (color, filled);
            this.width=width;
            this.length=length;
            this.color=color;
            this.filled=filled;
        }
        //for width
        public double getWidth(double width){
            return width;
        }

        public void setWidth(double width){
            this.width=width;
        }

        //for length
        public double getLength(double length){
            return length;
        }

        public void setLength(double length){
            this.length=length;
        }

        public double getArea(double width,double length){
            return (this.width*this.length);
        }

        public double getPerimeter(double width,double length){
            return (2*this.width+2*this.length);
        }

        public String toString(){
            String s = "A Rectangle with width " + width+" and
length "+length + " which is a subclass of" +
super.toString();
            return s;
        }
    }

    class Square extends Rectangle
    {

        public Square()
        {
            super();
            double side = 1.0;
```

```java
        }

        public Square(double side)
        {
            super(side, side);
            side =side;
        }

        public Square(double side, String color, boolean filled)
        {
            super(side, side, color, filled);
            side =side;
        }

        public double getSide()
        {
            return super.getWidth(10);
        }

        public void setSide(double side)
        {
            super.setLength(side);
            super.setWidth(side);
        }

        public double getArea()
        {
            return getSide()*getSide();
        }

        public double getPerimeter()
        {
            return 4*getSide();
        }


        public String toString()
        {
            return "A Square with side = " + getSide() + ",
    which is a subclass of " + super.toString();
        }
    }
```

Output:

```
Run: CirRect
"C:\Program Files\Java\jdk-16.0.2\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\In
Perimeter of circle is:31.41592653589793
Area of circle is:78.53981633974483
Perimeter of rectangle is:30.0
Area of rectangle is:50.0
Perimeter of square is:40.0
Area of square is:40.0
```