ASSIGNMENT-7

20BCSE50\_Kumar Jijnasu\_C1-08

1..

class Person6 {

    private String name;

    private String address;

    Person6()

    {

    }

    Person6(String n, String a)

    {

        name = n;

        address = a;

    }

    String getName()

    {

        return address;

    }

    String getAddress()

    {

        return address;

    }

    void setName(String n)

    {

        name = n;

    }

    void setAddress(String a)

    {

        address = a;

    }

    String ttoString()

    {

        return name+"("+address+")";

    }

}

class Student6 extends Person6

{

    private int numCourses;

    private String courses[];

    private int grades[];

    Student6()

    {

    }

    Student6(String n, String a)

    {

        setName(n);

        setAddress(a);

    }

    void addCourseGrade(String c[], int g[])

    {

        numCourses = c.length;

        courses = c;

        grades = g;

    }

    void printGrades()

    {

        System.out.println("Number of courses = "+numCourses);

        for(int i=0;i<grades.length;i++)

            System.out.println((i+1)+":\t"+courses[i]+" :- "+grades[i]);

    }

    double getAverageGrade()

    {

        int sum = 0;

        for(int i=0;i<numCourses;i++)

            sum += grades[i];

        return 1.0\*sum/numCourses;

    }

    String ttoString()

    {

        return "Student: "+super.ttoString();

    }

}

class Teacher6 extends Person6

{

    private int numCourses;

    private String courses[];

    Teacher6(String n,String a)

    {

        setName(n);

        setAddress(a);

    }

    boolean addCourse(String c)

    {

        for(int i=0;i<numCourses;i++)

            if(courses[i]==c)

                return false;

        numCourses += 1;

        String tcrs[] = new String[numCourses];

        for(int i=0;i<numCourses-1;i++)

            tcrs[i] = courses[i];

        tcrs[numCourses-1] = c;

        courses = tcrs;

        return true;

    }

    boolean removeCourse(String c)

    {

        for(int i=0;i<numCourses;i++)

            if(courses[i]==c)

            {

                int k=0;

                String tcrs[] = new String[numCourses-1];

                for(i=0;i<numCourses;i++)

                    if(courses[i]!=c)

                        tcrs[k++] = courses[i];

                return true;

            }

        return false;

    }

    String ttoString()

    {

        return "Teacher: "+super.ttoString();

    }

}

class personDriver

{

    public static void main(String[] args) {

        Student6 stu = new Student6("hemlo gmys","bbsr1");

        String c[] = {"crs A","crs B","crs C"};

        int g[] = {30,50,70};

        stu.addCourseGrade(c,g);

        stu.printGrades();

        System.out.println("Average : "+stu.getAverageGrade()+"\n"+stu.ttoString());

        Teacher6 th = new Teacher6("theacher one", "bbsr2");

        System.out.println(th.addCourse("course-1"));

        System.out.println(th.addCourse("course-2"));

        System.out.println(th.addCourse("course-1"));

        System.out.println(th.addCourse("course-3"));

        System.out.println(th.removeCourse("course-3"));

        System.out.println(th.removeCourse("course-5"));

        System.out.println(th.ttoString());

    }

}

2..

class Shape {

    private String color;

    private boolean filled;

    Shape()

    {

        color = "red";

        filled = true;

    }

    Shape(String c,boolean f)

    {

        color = c;

        filled = f;

    }

    String getColor()

    {

        return color;

    }

    void setColor(String c)

    {

        color = c;

    }

    boolean isFilled()

    {

        return filled;

    }

    void setFilled(boolean f)

    {

        filled = f;

    }

    String ttoString()

    {

        return "COLOR: "+color+", FILLED: "+filled;

    }

}

class Circle extends Shape

{

    private double radius;

    Circle()

    {

        radius = 1.0;

    }

    Circle(double r)

    {

        radius = r;

    }

    Circle(double r, String c, boolean f)

    {

        setColor(c);

        setFilled(f);

        radius = r;

    }

    double getRadius()

    {

        return radius;

    }

    void setRadius(double r)

    {

        radius = r;

    }

    double getArea()

    {

        return 3.14\*radius\*radius;

    }

    double getPerimeter()

    {

        return 2\*3.14\*radius;

    }

    String ttoString(){

        return "A Circle with radius = "+getRadius()+" which is a subclass of "+super.ttoString();

    }

}

class Rectangle extends Shape

{

    double width,length;

    Rectangle()

    {

        setWidth(1.0);

        setLength(1.0);

    }

    Rectangle(double width1,double length1)

    {

        setWidth(width1);

        setLength(length1);

    }

    double getWIDTH(){

        return width;

    }

    double getLENGTH(){

        return length;

    }

    void setWidth(double width1){

        width=width1;

    }

    void setLength(double length1){

        length=length1;

    }

    double getArea(){

        return width\*length;

    }

    double getPerimeter(){

        return 2\*(width+length);

    }

    String ttoString(){

        return "A Rectangle with width = "+getWIDTH()+" and length = "+getLENGTH()+" which is a subclass of "+super.ttoString();

    }

}

class Square extends Rectangle

{

    Square()

    {

        super();

    }

    Square(double side)

    {

        super(side,side);

    }

    String ttoString()

    {

        return "A Square with side = "+getWIDTH()+" which is a subclass of "+super.ttoString();

    }

    void setLength(double l)

    {

        super.setLength(l);

    }

    void setWidth(double w)

    {

        super.setWidth(w);

    }

}

class ptDriver

{

    public static void main(String[] args) {

*// SHAPE*

        Shape shp1 = new Shape();

        System.out.println(shp1.ttoString());

        Shape shp2 = new Shape("BLUE",false);

        System.out.println(shp2.ttoString());

*// CIRCLE*

        Circle c1 = new Circle();

        System.out.println(c1.ttoString());

        Circle c2 = new Circle(5.5);

        System.out.println(c2.ttoString());

*// RECTANGLE*

        Rectangle r1 = new Rectangle();

        System.out.println(r1.ttoString());

        Rectangle r2 = new Rectangle(2,3);

        System.out.println(r2.ttoString());

*// SQUARE*

        Square s1 = new Square();

        System.out.println(s1.ttoString());

        Square s2 = new Square(5);

        System.out.println(s2.ttoString());

    }

}

3..

*// import java.util.function.DoubleUnaryOperator;*

abstract class absShape{

    protected String color;

    protected boolean filled;

    absShape(){

        setColor("green");

        setFilled(true);

    }

    absShape(String color1,boolean filled1){

        setColor(color1);

        setFilled(filled1);

    }

    String getColor(){

        return color;

    }

    void setColor(String color1){

        color=color1;

    }

    boolean isFilled(){

        return filled;

    }

    void setFilled(boolean filled1){

        filled=filled1;

    }

    abstract String tostring();

    abstract double getArea();

    abstract double getPerimeter();

*// abstract double getRadius();*

}

class Circle extends absShape{

    protected double radius;

    Circle(){setRadius(0.0);}

    Circle(double radius1){

        setRadius(radius1);

    }

    Circle(double radius1,String color1,boolean filled1){

        setRadius(radius1);

        setColor(color1);

        setFilled(filled1);

    }

    double getRadius(){

        return radius;

    }

    void setRadius(double radius1){

        radius=radius1;

    }

    double getArea(){

        return 3.14\*radius\*radius;

    }

    double getPerimeter(){

        return 2\*3.14\*radius;

    }

    String tostring(){

        return "A Circle with radius ="+getRadius()+" which is a subclass of "+tostring();

    }

}

class Rectangle extends absShape{

    protected double width,length;

    Rectangle(){

        setWidth(0.0);

        setLength(0.0);

    }

    Rectangle(double width1,double length1){

        setWidth(width1);

        setLength(length1);

    }

    Rectangle(double width1,double length1,String color1,boolean filled1){

        setWidth(width1);

        setLength(length1);

        setColor(color1);

        setFilled(filled1);

    }

    double getWidth(){

        return width;

    }

    double getLength(){

        return length;

    }

    void setWidth(double width1){

        width=width1;

    }

    void setLength(double length1){

        length=length1;

    }

    double getArea(){

        return width\*length;

    }

    double getPerimeter(){

        return 2\*(width+length);

    }

    String tostring(){

        return "A Rectangle with width ="+getWidth()+" and length"+getLength()+" which is a subclass of "+tostring();

    }

}

class Square extends Rectangle{

    double side;

    Square(){}

    Square(double side){

        setSide(side);

    }

    Square(double side1,String color1,boolean filled1){

        setSide(side1);

        setColor(color1);

        setFilled(filled1);

    }

    double getSide(){

        return side;

    }

    void setSide(double side1){

        side=side1;

    }

    String tostring(){

        return "A Square with side ="+getSide()+" which is a subclass of "+tostring();

    }

    void setLength(double side)

    {

        super.setLength(side);

    }

    void setWidth(double side)

    {

        super.setWidth(side);

    }

}

class ptDriver1

{

    public static void main(String[] args) {

        absShape s1 = new Circle(5.5, "RED", false); *// Upcast Circle to Shape*

        System.out.println(s1); *// which version?*

        System.out.println(s1.getArea()); *// which version?*

        System.out.println(s1.getPerimeter()); *// which version?*

        System.out.println(s1.getColor());

        System.out.println(s1.isFilled());

*// System.out.println(s1.getRadius());*

        Circle c1 = (Circle)s1; *// Downcast back to Circle*

        System.out.println(c1);

        System.out.println(c1.getArea());

        System.out.println(c1.getPerimeter());

        System.out.println(c1.getColor());

        System.out.println(c1.isFilled());

        System.out.println(c1.getRadius());

*// absShape s2 = new absShape();*

        absShape s3 = new Rectangle(1.0, 2.0, "RED", false); *// Upcast*

        System.out.println(s3);

        System.out.println(s3.getArea());

        System.out.println(s3.getPerimeter());

        System.out.println(s3.getColor());

*// System.out.println(s3.getLength());*

        Rectangle r1 = (Rectangle)s3; *// downcast*

        System.out.println(r1);

        System.out.println(r1.getArea());

        System.out.println(r1.getColor());

        System.out.println(r1.getLength());

        absShape s4 = new Square(6.6); *// Upcast*

        System.out.println(s4);

        System.out.println(s4.getArea());

        System.out.println(s4.getColor());

*// System.out.println(s4.getSide());*

*// Take note that we downcast Shape s4 to Rectangle,*

*// which is a superclass of Square, instead of Square*

        Rectangle r2 = (Rectangle)s4;

        System.out.println(r2);

        System.out.println(r2.getArea());

        System.out.println(r2.getColor());

*// System.out.println(r2.getSide());*

        System.out.println(r2.getLength());

    }

}