**实验5接口与多态**

**一、实验时间**

2023年10月17日星期二

**二、实验目的**

1、理解接口、多态的概念、作用和意义；

2、掌握接口的定义及应用；

3、理解并掌握多态的实现机制；

4、掌握多态的应用。

**三、实验内容**

（1）定义接口IShape，其中包含方法：

double getArea(); //获取图形面积

double getCircum(); //获取图形周长

void moveTo(intposx,int posy); //移动图形到指定的位置

void zoom(double i); //对图形进行缩放，缩放比例i>0

（2）定义Point类，包含坐标x，y，定义其无参和带参构造方法、set和get方法。

（3）定义Circle类，要求实现IShape接口，定义半径radius，圆心位置posx、posy等属性；定义无参和带参构造方法；定义set和get方法；实现moveTo方法通过移动圆心的位置实现；实现zoom方法，主要是对半径的大小进行缩放；

（4）定义Triangle类，要求实现IShape接口，定义三个顶点；定义无参和带参构造方法；定义set和get方法；moveTo方法实现将三角形的中心点位置移动到指定的位置（假设中心点位置的x、y分别为三个顶点坐标的均值）；**实现zoom方法，主要是对任意两个顶点的坐标值进行缩放**；

（5）定义Rectangle类，要求实现IShape接口，定义左上角顶点、长、宽属性；定义无参和带参构造方法；定义set和get方法；moveTo方法实现将矩形的左上角位置移动到指定的位置；**实现zoom方法，主要是对长和宽的长度进行缩放**；

（6）定义ShapeTest类，定义一个ArrayList，添加两个Circle类对象、两个Triangle类对象、两个Rectangle类对象。定义：

方法：double printArea(IShapeishape); //打印输出面积，自定义打印信息格式；

方法：double printCircum(IShapeishape); //打印输出周长，自定义打印信息格式；

方法：doublemoveTo(IShapeishape);

方法：double zoom(double i);

a、通过moveTo方法，对上述6个对象分别进行移动，并输出相关信息；

b、通过printArea方法，将上述6个对象的面积打印输出；

c、通过printCircum方法，将上述6个对象的周长打印输出；

d、调用zoom方法，对上述6个对象进行缩放，然后通过printArea、printCircum方法依次重新打印各对象的面积和周长。

**四、实验过程**

1、实验原理分析（写出你采用的实验理论基础）（10分）

接口的使用，ArrayList动态数组的使用。

2、实验设计（写出实验的算法设计）（25分）

我设计了一个IShape的接口，除了按照要求设置的类外，为方便计算，我还三角形类在里面自行添加了两点之间求距离的方法。

3、实验过程（写出实验过程中遇到的问题及解决的思路和方法）（30分）

分文件编写。

４、实验源码（20分）

//IShape.java

package 源代码存储.Java.实验5.Code;

public interface IShape {

        double getArea();    //获取图形面积

        double getCircum();  //获取图形周长

        void moveTo(int posx,int posy);//移动图形到指定的位置

        void zoom(double i);    //对图形进行缩放，缩放比例i>0

}

//Cirecle.java

package 源代码存储.Java.实验5.Code;

public class Circle implements IShape{

    double redius;//半径

    double posx;

    double posy;

    public Circle()

    {

        redius =1;

        posx=0;

        posx=1;

    }//无参构造函数，默认值101

    public Circle(double r,double x,double y)

    {

        redius =r;

        posx=x;

        posy=y;

    }

    public void setX(double x)

    {

        posx=x;

    }

    public void setY(double y)

    {

        posy=y;

    }

    public void setR(double r)

    {

        redius=r;

    }

    public double getX(){

        return posx;

    }

    public double getY(){

        return posy;

    }

    public double getR(){

        return redius;

    }

    @Override

    public double getArea()

    {

        return Math.PI\*redius\*redius;

    }    //获取图形面积

    @Override

    public double getCircum()

    {

        return 2.0\*Math.PI\*redius;

    }  //获取图形周长

    @Override

    public void moveTo(int posx,int posy)

    {

        this.posx=posx;

        this.posy=posy;

        System.out.println("圆心被移动到了"+"("+posx+","+posy+")");

    }//移动图形到指定的位置

    @Override

    public void zoom(double i)

    {

    redius\*=i;

    }    //对图形进行缩放，缩放比例i>0

}

//Point.java

package 源代码存储.Java.实验5.Code;

public class Point implements IShape{

    double  x,y;

    public Point(){//无参

        x=0;

        y=0;

    }

    public Point(double x,double y){

        this.x=x;

        this.y=y;

    }

    public void setX(double x){

        this.x=x;

    }

    public void setY(double y){

        this.y=y;

    }

    public double getX(){

        return x;

    }

    public double getY(){

        return y;

    }

    @Override

    public double getArea()

    {

        return 0;

    }    //获取图形面积

    @Override

    public double getCircum()

    {

        return 0;

    }  //获取图形周长

    @Override

    public void moveTo(int posx,int posy)

    {

        x=posx;

        y=posy;

    }//移动图形到指定的位置

    @Override

    public void zoom(double i)

    {

    }    //对图形进行缩放，缩放比例i>0

}

//Rectangle.java

package 源代码存储.Java.实验5.Code;

public class Rectangle implements IShape{

    Point leftPoint;//左顶点

    double length;//长度

    double width;//宽度

    public Rectangle()

    {

        leftPoint=new Point(0,0);

        length=1;

        width=2;;

    }

    public Rectangle(int posx,int posy,int length,int width)

    {

        leftPoint=new Point (posx,posy);

        this.length=length;

        this.width=width;

    }

    @Override

    public double getArea()

    {

        return length\*width;

    }    //获取图形面积

    @Override

    public double getCircum()

    {

        return (length+width)\*2;

    }  //获取图形周长

    @Override

    public void moveTo(int posx,int posy)

    {

        leftPoint.x=posx;

        leftPoint.y=posy;

        System.out.println("左顶点被移动到了"+"("+posx+","+posy+")");

    }//移动图形到指定的位置

    @Override

    public void zoom(double i)

    {

        length=length\*i;

        width=width\*i;

    }    //对图形进行缩放，缩放比例i>0

}

//Triangle.java

package 源代码存储.Java.实验5.Code;

public class Triangle implements IShape{

    //设置三个点

    Point p1;

    Point p2;

    Point p3;

    double avx;//类中心点x

    double avy;//类中心点y

    public Triangle()

    {

        p1=new Point(0,0);

        p2=new Point(1,1);

        p3=new Point(2,2);

        avx=(0+1+2)\*1.0/3;

        avy=avx;

    }

    public Triangle(double p1x,double p1y,double p2x,double p2y,double p3x,double p3y)

    {

        p1=new Point(p1x,p1y);

        p2=new Point(p2x,p2y);

        p3=new Point(p3x,p3y);

        avx=(p1x+p2x+p3x)/3;

        avy=(p1x+p2x+p3x)/3;

    }

    public void set\_p1(double x,double y)

    {

        p1.setX(x);

        p1.setY(y);

        avx=(x+p2.x+p3.x)/3;

        avy=(y+p2.y+p3.y)/3;

    }

    public void set\_p2(double x,double y)

    {

        p2.setX(x);

        p2.setY(y);

        avx=(p1.x+p2.x+p3.x)/3;

        avy=(p1.y+p2.y+p3.y)/3;

    }

    public void set\_p3(double x,double y)

    {

        p3.setX(x);

        p3.setY(y);

        avx=(p1.x+p2.x+p3.x)/3;

        avy=(p1.y+p2.y+p3.y)/3;

    }

    //得到每个点的get函数

    public double get\_p1\_x(){

        return p1.getX();

    }

    public double get\_p2\_x(){

        return p2.getX();

    }

    public double get\_p3\_x(){

        return p3.getX();

    }

    public double get\_p1\_y(){

        return p1.getY();

    }

    public double get\_p2\_y(){

        return p2.getY();

    }

    public double get\_p3\_y(){

        return p3.getY();

    }

    @Override

    public double getArea()

    {

        double a=Instance(p1, p2);

        double b=Instance(p2, p3);

        double c=Instance(p3, p1);

        //海伦公式1/4\*sqrt((a+b+c)(a+b-c)(a+c-b)(b+c-a))

        return 0.25\*Math.sqrt((a+b+c)\*(a+b-c)\*(a+c-b)\*(b+c-a));

    }    //获取图形面积

    @Override

    public double getCircum()

    {

        return Instance(p1,p2)+Instance(p2, p3)+Instance(p1, p3);

    }  //获取图形周长

    @Override

    public void moveTo(int posx,int posy)

    {

        avx=posx;

        avy=posy;

        System.out.println("三角形中心被移动到了"+"("+posx+","+posy+")");

    }//移动图形到指定的位置

    @Override

    public void zoom(double i)

    {

        //任意缩放，我选择p1 p2;

        p1.x=p1.x\*i;

        p1.y=p1.y\*i;

        p2.x=p2.x\*i;

        p2.y=p2.y\*i;

    }    //对图形进行缩放，缩放比例i>0

    public double Instance(Point p1,Point p2)

    {

        return Math.sqrt(Math.pow((p1.x-p2.x),2)+Math.pow((p1.y-p2.y),2));

    }

}

//ShapeTest.java

package 源代码存储.Java.实验5.Code;

import java.util.ArrayList;

public class ShapeTest {

public static void moveTo(IShape shape, int x, int  y) {

    shape.moveTo(x, y);

}

public static void printArea(IShape shape) {

    System.out.println("Shape area: " + shape.getArea());

}

public static void printCircum(IShape shape) {

    System.out.println("Shape circumference: " + shape.getCircum());

}

public static void zoom(IShape shape, double factor) {

    shape.zoom(factor);

}

    public static void main(String[] args) {

    ArrayList<IShape> shapes = new ArrayList<>();

    shapes.add(new Circle(5,0,0));

    shapes.add(new Circle(10,0,0));

    shapes.add(new Triangle(1,1,2,2,3,4));

    shapes.add(new Triangle(0,3,4,0,0,0));

    shapes.add(new Rectangle(0,0,2, 5));

    shapes.add(new Rectangle(0,0,3,7));

    for(IShape i:shapes)

    {

        printCircum(i);

    }

    for(IShape i:shapes)

    {

        printArea(i);

    }

    //统一移动

    for(IShape i:shapes)

    {

        moveTo(i,0,2);

    }

    //统一缩放

    for(IShape i:shapes)

    {

        zoom(i, 10);

    }

    System.out.println("扩缩后");

    for(IShape i:shapes)

    {

        printCircum(i);

    }

    for(IShape i:shapes)

    {

        printArea(i);

    }

}

}

５、运行结果（要求对结果分析，分别对输入出错和输入正确的结果进行分析）（10分）

Shape circumference: 31.41592653589793

Shape circumference: 62.83185307179586

Shape circumference: 7.255832815336873

Shape circumference: 12.0

Shape circumference: 14.0

Shape circumference: 20.0

Shape area: 78.53981633974483

Shape area: 314.1592653589793

Shape area: 0.5000000000000008

Shape area: 6.0

Shape area: 10.0

Shape area: 21.0

圆心被移动到了(0,2)

圆心被移动到了(0,2)

三角形中心被移动到了(0,2)

三角形中心被移动到了(0,2)

左顶点被移动到了(0,2)

左顶点被移动到了(0,2)

扩缩后

Shape circumference: 314.1592653589793

Shape circumference: 628.3185307179587

Shape circumference: 46.706915140881335

Shape circumference: 120.0

Shape circumference: 140.0

Shape circumference: 200.0

Shape area: 7853.981633974483

Shape area: 31415.926535897932

Shape area: 4.999999999999977

Shape area: 600.0

Shape area: 1000.0

Shape area: 2100.0

6、实验总结（根据实验的过程和结果分析写出你对实验原理的理解）（5分）

通过本次实验，我对接口和多态有了更深入的理解和应用。以下是对本次实验的总结：

首先，接口是一个重要的编程概念，它定义了一个契约或规范，供实现该接口的类参考和实现。在本次实验中，我们定义了IShape接口，并明确了它的四个方法：getArea()、getCircum()、moveTo()和zoom()。通过这个接口，我们可以期望所有实现该接口的类都有这四个方法，并且能按照我们期望的方式进行操作。

在IShape接口的基础上，我们实现了几个具体的类，包括Circle、Triangle、Rectangle等。这些类都实现了IShape接口，也就实现了接口中定义的方法。通过这些类的实现，我们可以看到多态的体现和应用。多态是指不同对象对同一消息做出不同的响应。在我们的代码中，同样是调用moveTo()方法，不同的类做出的动作却不同。例如，Circle和Rectangle类的moveTo()方法移动的是物体的中心，而Triangle类的moveTo()方法则是将三角形的中心移动到指定位置。

在实验中，我们还编写了一个ShapeTest类来进行多个对象的操作。这个类里面定义了四个方法：printArea()、printCircum()、moveTo()和zoom()。通过这个类，我们可以对所有实现的IShape接口的类进行同样的操作。例如，我们可以调用zoom()方法对所有对象进行缩放，然后再通过printArea()和printCircum()方法打印出缩放后的面积和周长。这种统一的操作方式体现了面向对象编程的封装性和多态性。

通过本次实验，我深刻理解了接口和多态的概念、作用和意义。接口为我们提供了一种方式来定义一组通用的方法，以供实现该接口的类参考和实现。多态则为我们提供了一种方式来使得不同类型的对象能够以一种统一和简洁的方式进行处理。此外，我也学会了如何定义和使用接口，如何通过继承和实现接口来创建子类，并学会了如何利用多态性对不同的对象进行相同的操作。

总的来说，这次实验提高了我的编程技能和理解能力，使我更加熟悉了接口和多态的使用。在未来的学习和工作中，我将更加注重这些概念的应用，以提高我的编程能力和代码的可读性和可维护性。

评分表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 项目 | 总分 | 评分标准 | 实际得分 |
| 1 | 实验原理分析 | 10 | 对实验全过程涉及的知识点应用进行有效分析 |  |
| 2 | 实验设计 | 25 | 类的封装机制、继承机制、多态机制、异常机制的设计（10分），类的属性成员的设计（5分），类的方法成员的设计（5分），测试部分输入的设计（3分）、测试部分输出的设计（2分） |  |
| 3 | 实验过程 | 30 | 对遇到的问题进行分析（10分），解决思路（5分）、方案（5分）、过程（10分）， |  |
| 4 | 实验源码 | 20 | 实验源码的书写规范性（2分）、标识符的命名规范（2分）、是否有注释（2分）；代码的正确性（14分） |  |
| 5 | 运行结果 | 10 | 是否有运行截图（5分），是否对结果进行分析（5分） |  |
| 6 | 实验总结 | 5 | 对实验是否进行了有效总结，主要体现在对知识点的理解，实验过程的总结、知识点应用总结 |  |
| 总分 |  | | | |
| 实验评价 |  | | | |