

**JAVA实验报告**

**学院： 计算机科学学院**

**班级：计算机科学与技术1802班**

**姓名： 段星雨**

**目录**

[13.6 1](#_Toc40723828)

[1. 实验题目 1](#_Toc40723829)

[2. 实验思路 1](#_Toc40723830)

[3. 目录框架 1](#_Toc40723831)

[4. 源代码 1](#_Toc40723832)

[Circle.java 1](#_Toc40723833)

[ComparableCircle.java 3](#_Toc40723834)

[GeometricObject.java 3](#_Toc40723835)

[Text.java 4](#_Toc40723836)

[5. UML图 5](#_Toc40723837)

[6. 运行结果 5](#_Toc40723838)

[13.7 6](#_Toc40723839)

[1. 实验题目 6](#_Toc40723840)

[2. 实验思路 6](#_Toc40723841)

[3. 目录框架 6](#_Toc40723842)

[4. 源代码 7](#_Toc40723843)

[Squar.java 7](#_Toc40723844)

[Colorable.java 8](#_Toc40723845)

[GeometricObject.java 8](#_Toc40723846)

[Text.java 8](#_Toc40723847)

[5. UML图 9](#_Toc40723848)

[6. 运行结果 9](#_Toc40723849)

# 13.6

## 1. 实验题目

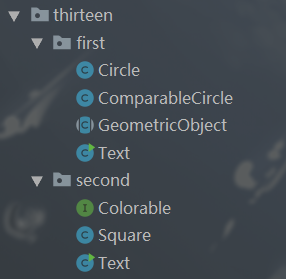
(The ComparableCircle class) Define a class named ComparableCircle that extends Circle and implements Comparable. Draw the UML diagram and implement the compareTo method to compare the circles on the basis of area. Write a test class to find the larger of two instances of ComparableCircle objects.

( ComparableCircle 类）创建名为ComparableCircle的类，它继承自Circle 类，并实现 Comparable 接口。画出UML图并实现 compareTo 方法，使其根据面积比较两个圆 测试程序求出 ComparableCircle 对象的两个实例中的较大者。

## 2. 实验思路

首先创建ComparableCircle和Circle类，然后实现ComparableCircle中的Comparable接口，然后重写接口的方法和抽象类中的抽象方法，最后创建两个实例来测试Comparable方法。

## 3. 目录框架



## 4. 源代码

### Circle.java

package thirteen.first;

public class Circle extends GeometricObject {

private double radius;

public Circle() {

}

public Circle(double radius) {

this.radius = radius;

}

public Circle(double radius, String color, boolean filled) {

this.radius = radius;

setColor(color);

setFilled(filled);

}

public double getRadius() {

return radius;

}

public void setRadius(double radius) {

this.radius = radius;

}

public double getDiameter() {

return 2 \* radius;

}

@Override

public double getArea() {

return radius \* radius \* Math.PI;

}

@Override

public double getPerimeter() {

return 2 \* radius \* Math.PI;

}

@Override

public String toString() {

return "Circle{" +

"radius=" + radius +

", color=" + getColor() +

", filled=" + isFilled() +

'}';

}

}

### ComparableCircle.java

package thirteen.first;

public class ComparableCircle extends Circle implements Comparable<ComparableCircle> {

@Override

public int compareTo(ComparableCircle o) {

if (getArea() > o.getArea()) return 1;

else if (getArea() < o.getArea()) return -1;

else return 0;

}

public ComparableCircle(double radius, String color, boolean filled) {

super(radius, color, filled);

}

}

### GeometricObject.java

package thirteen.first;

import java.util.Date;

public abstract class GeometricObject {

private String color = "white";

private boolean filled;

private java.util.Date dateCreated;

protected GeometricObject() {

dateCreated = new Date();

}

protected GeometricObject(String color, boolean filled) {

dateCreated = new Date();

this.color = color;

this.filled = filled;

}

public String getColor() {

return color;

}

public void setColor(String color) {

this.color = color;

}

public boolean isFilled() {

return filled;

}

public void setFilled(boolean filled) {

this.filled = filled;

}

public Date getDateCreated() {

return dateCreated;

}

@Override

public String toString() {

return "created on " + dateCreated + "\ncolor: " + color + " and filled: " + filled;

}

public abstract double getArea();

public abstract double getPerimeter();

}

### Text.java

package thirteen.first;

public class Text {

public static void main(String[] args) {

ComparableCircle cc1 = new ComparableCircle(7.5, "red", true);

ComparableCircle cc2 = new ComparableCircle(6, "black", false);

int result = cc1.compareTo(cc2);

System.out.println("两圆的规格为：");

System.out.println(cc1);

System.out.println(cc2);

if (result == 1) System.out.println("第一个面积圆大");

else if (result == -1) System.out.println("第二个面积圆大");

else System.out.println("面积同样大");

}

}

## 5. UML图

GeometricObject

+ ComparableCircle()

Circle

-radius: double

+Circle()

+Circle(radius: double)

+Circle(radius: double, color: string, filled: boolean)

+getRadius(): double

+setRadius(radius: double): void +getDiameter(): double

+toString(): String

*GeometricObject*

-color: String

-filled: boolean

-dateCreated: java.util.Date

#GeometricObject()

#GeometricObject(color: string, filled: boolean)

+getColor(): String

+setColor(color: String): void +isFilled(): boolean

+setFilled(filled: boolean): void +getDateCreated():Date

+toString(): String

*+getArea(): double*

*+getPerimeter(): double*

«interface»

*java.lang.Comparable<ComparableRectangle>*

*+compareTo(o: ComparableRectangle): int*

## 6. 运行结果

# 13.7

## 1. 实验题目

(The Colorable interface) Design an interface named Colorable with a void method named howToColor(). Every class of a colorable object must implement the Colorable interface. Design a class named Square that extends GeometricObject and implements Colorable. Implement howToColor to display the message Color all four sides.

Draw a UML diagram that involves Colorable, Square, and GeometricObject. Write a test program that creates an array of five GeometricObjects. For each object in the array, display its area and invoke its howToColor method if it is colorable.

(可着色接口Colorable) 设计一个名为Colorable的接口，其中有名为howToColor() 的 void方法。可着色对象的每个类必须实现Colorable 接口。设计一个名为 Square的类，继承自CeometricObject 类并实现Colorable 接口。实现howToColor 方法，显示一个消息 Color all four sides (给所有的四条边着色）。

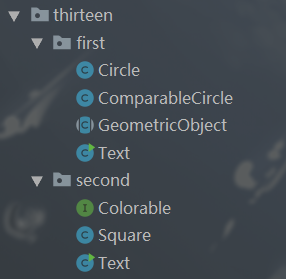
画出包含 Colorable、Square 和CeometricObject的UML图。编写一个测试程序，创建有五个CeometricObject 对象的数组。对于数组中的每个对象而言，如果对象是可着色的， 那就调用howToColor 方法。

## 2. 实验思路

创建Colorable接口，然后创建Square类继承CeometricObject实现Colorable，之后重写接口的方法和抽象类中的抽象方法。

可着色的判定采用转型的方法，将CeometricObject的元素向下转型为可以实现howToColor()方法的Square类，如果没有捕获到异常，继续着色（使用howToColor()），如果捕获错误则不着色（不使用howToColor()）。

## 3. 目录框架



## 4. 源代码

### Squar.java

package thirteen.second;

import thirteen.first.GeometricObject;

public class Square extends GeometricObject implements Colorable {

private double len;

public Square(double len) {

this.len = len;

}

public Square(String color, boolean filled, double len) {

setColor(color);

setFilled(filled);

this.len = len;

}

public double getLen() {

return len;

}

public void setLen(double len) {

this.len = len;

}

@Override

public void howToColor() {

System.out.println("Color all four sides");

}

@Override

public double getArea() {

return len \* len;

}

@Override

public double getPerimeter() {

return 4 \* len;

}

@Override

public String toString() {

return "Square{" +

"len=" + len +

", color=" + getColor() +

", filled=" + isFilled() +

'}';

}

}

### Colorable.java

package thirteen.second;

public interface Colorable {

void howToColor();

}

### GeometricObject.java

见13.6源代码。

### Text.java

package thirteen.second;

import thirteen.first.Circle;

import thirteen.first.GeometricObject;

public class Text {

public static void main(String[] args) {

GeometricObject[] goArr = {

new Circle(7.5, "red", true),

new Square("blue", false, 8),

new Circle(6.9, "black", true),

new Square("pink", true, 2.7),

new Circle(6.8, "purple", false),

};

for (GeometricObject go : goArr) {

//多态，找最新实现的toString

System.out.println(go);

//转型报错之后不调用howToColor()

try {

Square temS = (Square) go;

temS.howToColor();

} catch (ClassCastException e) {

}

System.out.println();

}

}

}

## 5. UML图

«interface»

*Colorable*

*+howToColor(): void*

Square

-len: double

+Square()

+Square(radius: double)

+Square(color: string, filled: boolean, len: double)

+getLen(): double

+setLen(len: double): void

+toString(): String

*GeometricObject*

-color: String

-filled: boolean

-dateCreated: java.util.Date

#GeometricObject()

#GeometricObject(color: string, filled: boolean)

+getColor(): String

+setColor(color: String): void +isFilled(): boolean

+setFilled(filled: boolean): void +getDateCreated():Date

+toString(): String

*+getArea(): double*

*+getPerimeter(): double*

## 6. 运行结果