13.6

public class TestCom{

public static void main(String args[]){

ComparableCricle cricle1=new ComparableCricle(5.0);

ComparableCricle cricle2=new ComparableCricle(4.0);

int result=cricle1.compareTo(cricle2);

if(result>0){

System.out.println("object1>object2");

}

else if(result<0){

System.out.println("object1<object2");

}

else

System.out.println("object1=object2");

}

}

interface CompareTo{

public int compareTo(ComparableCricle o);

}

class Cricle{

protected double radius=1.0;

Cricle(){};

Cricle(double radius){

this.radius=radius;

}

public double getArea(){

return Math.PI\*radius\*radius;

}

}

class ComparableCricle extends Cricle implements CompareTo{

ComparableCricle(double radius){

super.getArea();

this.radius=radius;

}

@Override

public int compareTo(ComparableCricle o){

if(getArea()>o.getArea()){

return 1;

}

else if(getArea()<o.getArea()){

return -1;

}

else

return 0;

}

}

13.7

UML图

|  |
| --- |
| GeometricObject |

|  |
| --- |
| *Colorable* |
| +howToColor: void |

|  |
| --- |
| Square |
|  |
| +Square()  + Square(color：Strin, filled：boolean)  +howToColor: void |

package java;

import java.lang.Comparable;

import java.util.ArrayList;

import java.util.Scanner;

public class Java {

public static void main(String[] args) {

Scanner input = new Scanner(System.in);

ArrayList<Square> list= new ArrayList<>();

for(int i = 0; i < list.size(); i++)

{

list.add(new Square(input.toString(), input.nextBoolean()));

}

for(int i = 0; i < list.size(); i++)

{

if(list.get(i).isFilled())

{

list.get(i).howToColor();

}

}

}

}

class GeometricObject

{

private String color = "white";

private boolean filled;

private java.util.Date dateCreated;

public GeometricObject()

{

dateCreated = new java.util.Date();

}

public GeometricObject(String color,boolean filled)

{

dateCreated = new java.util.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 java.util.Date getDateCreated()

{

return dateCreated;

}

@Override

public String toString()

{

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

}

}

interface Colorable

{

public abstract void howToColor();

}

class Square extends GeometricObject implements Colorable

{

public Square()

{

super();

}

public Square(String color,boolean filled)

{

super(color, filled);

}

@Override

public void howToColor()

{

if(isFilled())

{

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

}

}

}