13.1

UML图：

|  |
| --- |
| *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():java.util.Date  +toString():String  *+getArea():double*  *+getPerimeter():double* |

|  |
| --- |
| Triangle |
| -a:double  -b:doube  -c:double |
| +Triangle()  +Triangle(aa:double,bb:double,cc:double)  +getA():double  +setA(aa:double):void  +getB():double  +setB(bb:double):void  +getC():double  +setC(cc:double):void  +getArea():double  +getPerimeter():double |

代码：

**import** java.util.Scanner;

**public** **class** one {

**public** **static** **void** main(String[] args)

{

System.***out***.println("Enter the number:");

Scanner input=**new** Scanner(System.***in***);

**double** a=input.nextDouble();

**double** b=input.nextDouble();

**double** c=input.nextDouble();

Triangle x=**new** Triangle(a,b,c);

System.***out***.println("the area: "+x.getArea()+",the perimeter: "+x.getPerimeter()+","+x.toString());

}

}

**abstract** **class** GeometricObject {

**private** String color = "white";

**private** **boolean** filled;

**private** java.util.Date dateCreated;

**protected** GeometricObject() {

dateCreated = **new** java.util.Date();

}

**protected** 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;

}

**public** String toString() {

**return** "created on " + dateCreated + "\ncolor: " + color +

" and filled: " + filled;

}

**public** **abstract** **double** getArea();

**public** **abstract** **double** getPerimeter();

}

**class** Triangle **extends** GeometricObject{

**private** **double** a;

**private** **double** b;

**private** **double** c;

**public** Triangle() {

}

**public** Triangle(**double** aa,**double** bb,**double** cc) {

a=aa;

b=bb;

c=cc;

}

**public** **double** getA()

{

**return** a;

}

**public** **void** setA(**double** aa)

{

a=aa;

}

**public** **double** getB()

{

**return** b;

}

**public** **void** setB(**double** bb)

{

b=bb;

}

**public** **double** getC()

{

**return** c;

}

**public** **void** setC(**double** cc)

{

c=cc;

}

**public** **double** getArea()

{

**double** area;

**double** s = (a + b + c) / 2;

area = Math.*sqrt*(s \* (s - a) \* (s - b) \* (s - c));

**return** area;

}

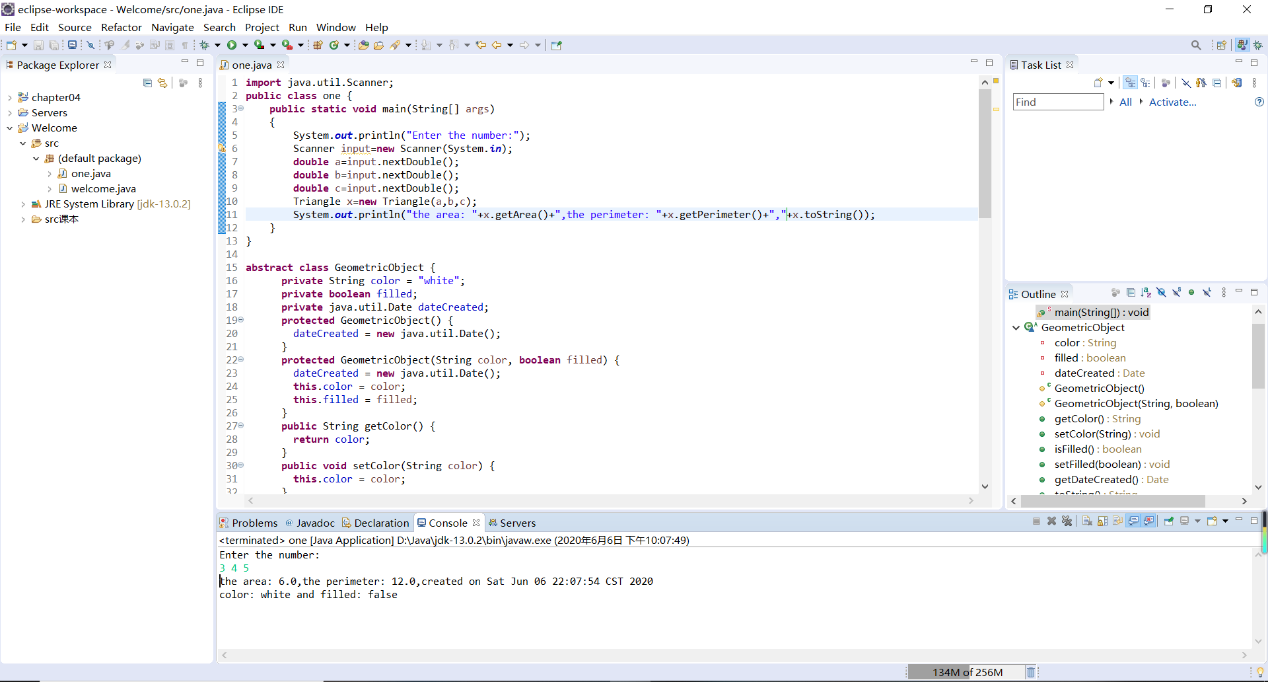
**public** **double** getPerimeter() {

**return** a+b+c;

}

}

截图：



13.5

UML图：

|  |
| --- |
| *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():java.util.Date  +toString():String  *+getArea():double*  *+getPerimeter():double*  +compareTo(a:GeometricObject):int  +max(a: GeometricObject,b: GeometricObject): GeometricObject |

|  |
| --- |
| Circle |
| -radius:double |
| +Circle()  +Circle(radius:double)  +getRadius():double  +setRadius(radius:double):void  +getDiameter():double  +printCircle():void |

|  |
| --- |
| Rectangle |
| -width:double  -height:double |
| +Rectangle()  +Rectangle(width:double,height:double)  +getWidth():double  +setWidth(width:double):void  +getHeight():double  +setHeight(height:double):void |

代码：

**public** **class** two {

**public** **static** **void** main(String[] args)

{

Circle x1=**new** Circle(5);

Circle x2=**new** Circle(4);

Rectangle y1=**new** Rectangle(3,4);

Rectangle y2=**new** Rectangle(4,5);

System.***out***.println("the max of the two circles is: ");

System.***out***.println(GeometricObject.*max*(x1, x2)==x1?"x1":"x2");

System.***out***.println("the max of the two rectangles is: ");

System.***out***.println(GeometricObject.*max*(y1, y2)==y1?"y1":"y2");

}

}

**abstract** **class** GeometricObject **implements** Comparable<GeometricObject> {

**private** String color = "white";

**private** **boolean** filled;

**private** java.util.Date dateCreated;

**protected** GeometricObject() {

dateCreated = **new** java.util.Date();

}

**protected** 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;

}

**public** String toString() {

**return** "created on " + dateCreated + "\ncolor: " + color +

" and filled: " + filled;

}

**public** **abstract** **double** getArea();

**public** **abstract** **double** getPerimeter();

**public** **int** compareTo(GeometricObject a)

{

**if**(getArea()>a.getArea())

**return** 1;

**else** **if**(getArea()<a.getArea())

**return** -1;

**else**

**return** 0;

}

**public** **static** GeometricObject max(GeometricObject a,GeometricObject b)

{

**if**(a.compareTo(b)==1)

**return** a;

**else** **if**(a.compareTo(b)==-1)

**return** b;

**else**

**return** a;

}

}

**class** Circle **extends** GeometricObject {

**private** **double** radius;

**public** Circle() {

}

**public** Circle(**double** radius) {

**this**.radius = radius;

}

**public** **double** getRadius() {

**return** radius;

}

**public** **void** setRadius(**double** radius) {

**this**.radius = radius;

}

@Override

**public** **double** getArea() {

**return** radius \* radius \* Math.***PI***;

}

**public** **double** getDiameter() {

**return** 2 \* radius;

}

**public** **double** getPerimeter() {

**return** 2 \* radius \* Math.***PI***;

}

**public** **void** printCircle() {

System.***out***.println("The circle is created " + getDateCreated() +

" and the radius is " + radius);

}

}

**class** Rectangle **extends** GeometricObject {

**private** **double** width;

**private** **double** height;

**public** Rectangle() {

}

**public** Rectangle(**double** width, **double** height) {

**this**.width = width;

**this**.height = height;

}

**public** **double** getWidth() {

**return** width;

}

**public** **void** setWidth(**double** width) {

**this**.width = width;

}

**public** **double** getHeight() {

**return** height;

}

**public** **void** setHeight(**double** height) {

**this**.height = height;

}

@Override

**public** **double** getArea() {

**return** width \* height;

}

@Override

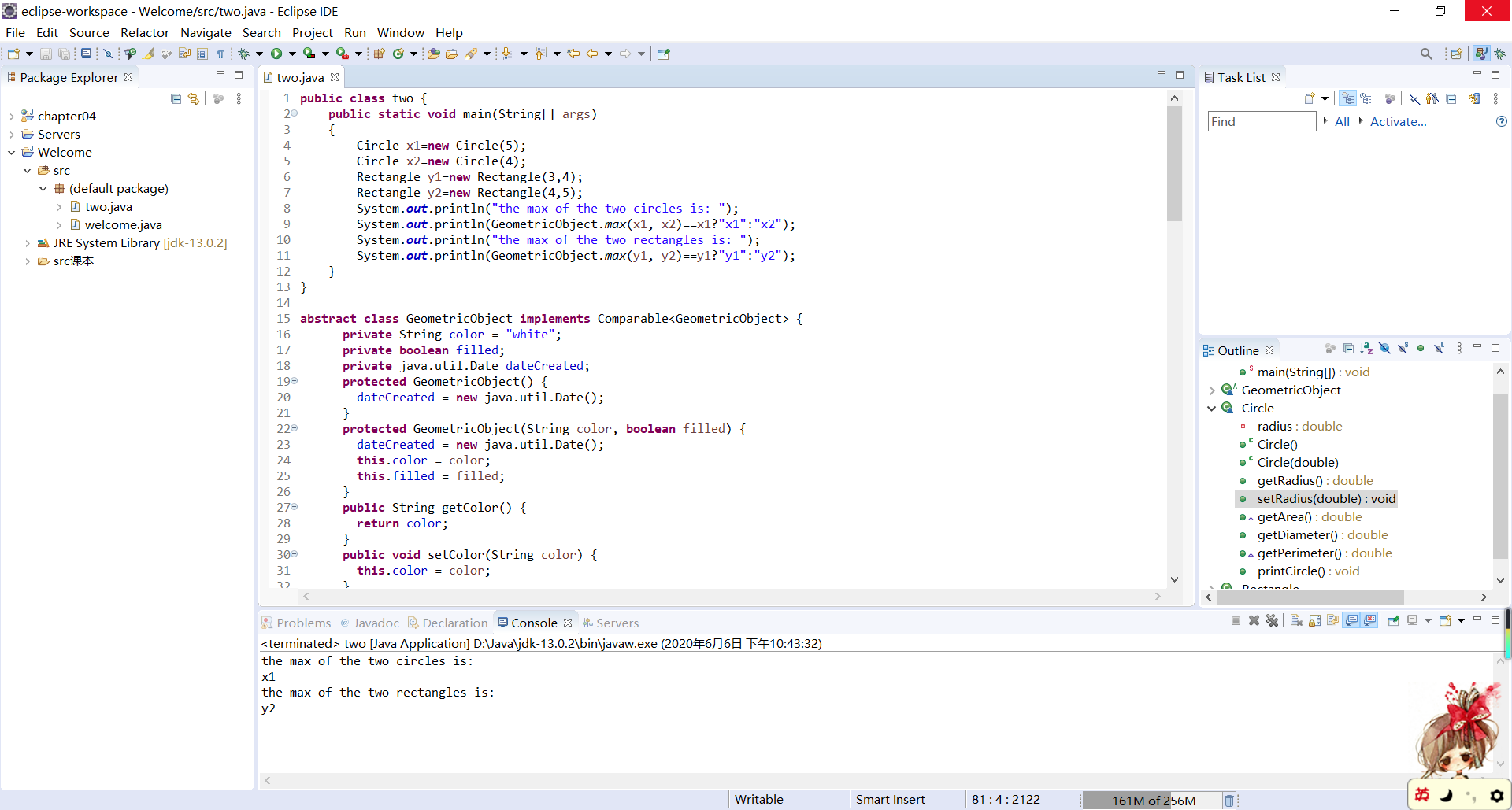
**public** **double** getPerimeter() {

**return** 2 \* (width + height);

}

}

截图：



13.9

UML图：

|  |
| --- |
| *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():java.util.Date  +toString():String  *+getArea():double*  *+getPerimeter():double* |

|  |
| --- |
| 《interface》  Java.lang.Comparable<Circle> |
| +compareTo(a:Circle):int |

|  |
| --- |
| Circle |
| -radius:double |
| +Circle()  +Circle(radius:double)  +getRadius():double  +setRadius(radius:double):void  +getDiameter():double  +printCircle():void  +compareTo(a:Circle):int  +equals(o:Object):boolean |

代码：

**public** **class** three {

**public** **static** **void** main(String[] args)

{

Circle x=**new** Circle(3);

Circle y=**new** Circle(5);

System.***out***.println("the result of comparison is: "+x.compareTo(y));

System.***out***.println("x equals to y? "+x.equals(y));

}

}

**abstract** **class** GeometricObject {

**private** String color = "white";

**private** **boolean** filled;

**private** java.util.Date dateCreated;

**protected** GeometricObject() {

dateCreated = **new** java.util.Date();

}

**protected** 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;

}

**public** String toString() {

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

}

**public** **abstract** **double** getArea();

**public** **abstract** **double** getPerimeter();

}

**class** Circle **extends** GeometricObject **implements** Comparable<Circle> {

**private** **double** radius;

**public** Circle() {

}

**public** Circle(**double** radius) {

**this**.radius = radius;

}

**public** **double** getRadius() {

**return** radius;

}

**public** **void** setRadius(**double** radius) {

**this**.radius = radius;

}

@Override

**public** **double** getArea() {

**return** radius \* radius \* Math.***PI***;

}

**public** **double** getDiameter() {

**return** 2 \* radius;

}

**public** **double** getPerimeter() {

**return** 2 \* radius \* Math.***PI***;

}

**public** **void** printCircle() {

System.***out***.println("The circle is created " + getDateCreated() + " and the radius is " + radius);

}

**public** **int** compareTo(Circle a) {

**if** (getArea() > a.getArea())

**return** 1;

**else** **if** (getArea() < a.getArea())

**return** -1;

**else**

**return** 0;

}

**public** **boolean** equals(Object o) {

**if** (o **instanceof** Circle)

**return** radius == ((Circle) o).radius;

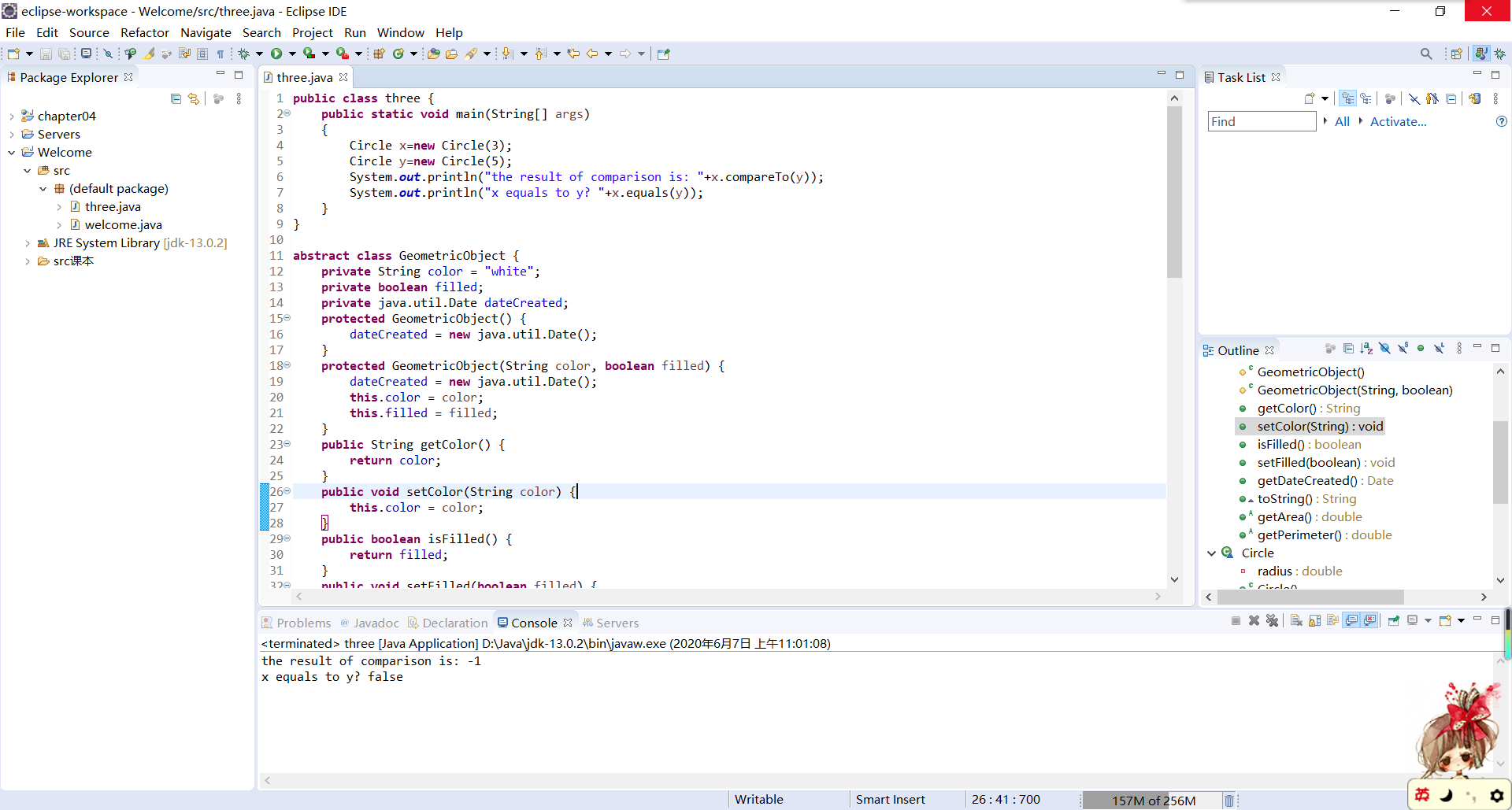
**else**

**return** **this** == o;

}

}

截图：



13.17

代码：

**import** java.util.Scanner;

**public** **class** four {

**public** **static** **void** main(String[] args)

{

Scanner input=**new** Scanner(System.***in***);

System.***out***.println("Enter the first complex number:");

**double** a=input.nextDouble();

**double** b=input.nextDouble();

System.***out***.println("Enter the second complex number:");

**double** c=input.nextDouble();

**double** d=input.nextDouble();

Complex x=**new** Complex(a,b);

Complex y=**new** Complex(c,d);

System.***out***.println("("+x+") + ("+y+") ="+x.add(y));

System.***out***.println("("+x+") - ("+y+") ="+x.substract(y));

System.***out***.println("("+x+") \* ("+y+") ="+x.multiply(y));

System.***out***.println("("+x+") / ("+y+") ="+x.divide(y));

System.***out***.println("|("+x+")| = "+x.abs());

}

}

**class** Complex **extends** Number **implements** Cloneable{

**private** **double** realpart=0;

**private** **double** imaginarypart=0;

**public** Complex()

{

}

**public** Complex(**double** a)

{

realpart=a;

}

**public** Complex(**double** a,**double** b)

{

realpart=a;

imaginarypart=b;

}

**public** **double** getRealpart()

{

**return** realpart;

}

**public** **double** getImaginarypart()

{

**return** imaginarypart;

}

**public** Complex add(Complex x)

{

**double** a=realpart+x.realpart;

**double** b=imaginarypart+x.imaginarypart;

**return** **new** Complex(a,b);

}

**public** Complex substract(Complex x)

{

**double** a=realpart-x.realpart;

**double** b=imaginarypart-x.imaginarypart;

**return** **new** Complex(a,b);

}

**public** Complex multiply(Complex x)

{

**double** a=realpart\*x.realpart-imaginarypart\*x.imaginarypart;

**double** b=imaginarypart\*x.realpart+realpart\*x.imaginarypart;

**return** **new** Complex(a,b);

}

**public** Complex divide(Complex x)

{

**double** a=(realpart\*x.realpart+imaginarypart\*x.imaginarypart)/(x.realpart\*x.realpart+x.imaginarypart\*x.imaginarypart);

**double** b=(imaginarypart\*x.realpart-realpart\*x.imaginarypart)/(x.realpart\*x.realpart+x.imaginarypart\*x.imaginarypart);

**return** **new** Complex(a,b);

}

**public** **double** abs()

{

**return** Math.*sqrt*(realpart\*realpart+imaginarypart\*imaginarypart);

}

**public** String toString()

{

**if**(imaginarypart==0)

**return** ""+realpart;

**else**

**return** realpart+" + "+imaginarypart+"i";

}

**public** Object clone() **throws** CloneNotSupportedException

{

**return** **super**.clone();

}

**public** **int** intValue()

{

**return** (**int**)doubleValue();

}

**public** **float** floatValue()

{

**return** (**float**)doubleValue();

}

**public** **double** doubleValue()

{

**return** realpart+imaginarypart\*Math.*sqrt*(-1);

}

**public** **long** longValue()

{

**return** (**long**)doubleValue();

}

}

截图：

