

9.1

UML类图：

|  |
| --- |
| Rectangle |
| +width:double  +height:double |
| +Rectangle()  +Rectangle(double,double)  +getArea():double  +getPerimeter():double |

代码：

**import** java.util.Scanner;

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

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

Rectangle A = **new** Rectangle(4, 40);

Rectangle B = **new** Rectangle(3.5, 35.9);

System.***out***.println("the first Rectangle: " + "width: " + A.width + ",height: " + A.height + ",Area: "

+ A.getArea() + ",Perimeter:" + A.getPerimeter());

System.***out***.println("the second Rectangle: " + "width: " + B.width + ",height: " + B.height + ",Area: "

+ B.getArea() + ",Perimeter:" + B.getPerimeter());

}

}

**class** Rectangle {

**public** **double** width = 1;

**public** **double** height = 1;

**public** Rectangle() {

}

**public** Rectangle(**double** w, **double** h) {

width = w;

height = h;

}

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

**return** width \* height;

}

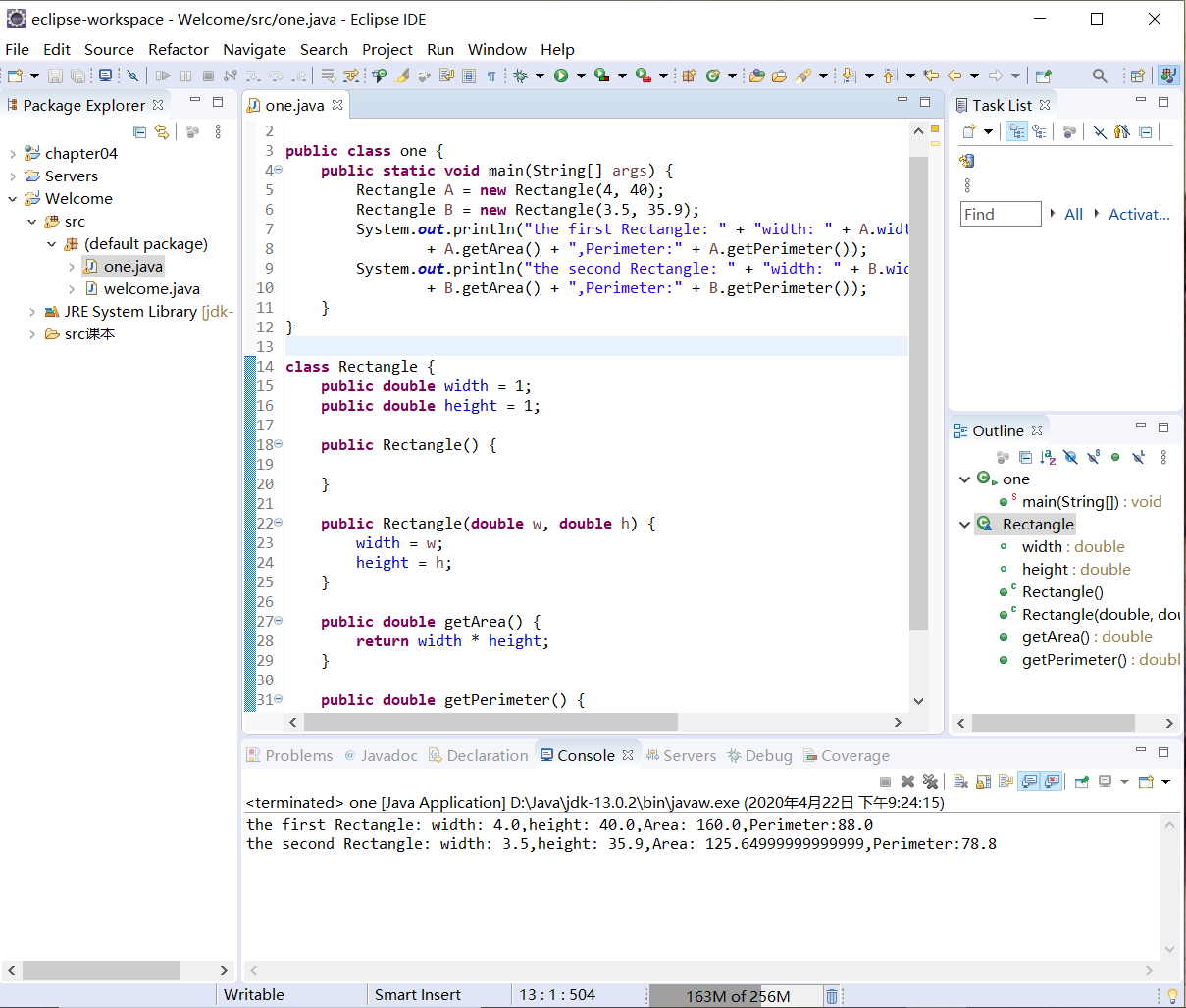
**public** **double** getPerimeter() {

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

}

}

截图：



9.4

代码：

**import** java.util.Random;

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

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

{

Random random1=**new** Random(1000);

**for**(**int** i=0;i<50;i++)

{

System.***out***.print(random1.nextInt(100));

**if**((i+1)%10==0)

System.***out***.println();

**else**

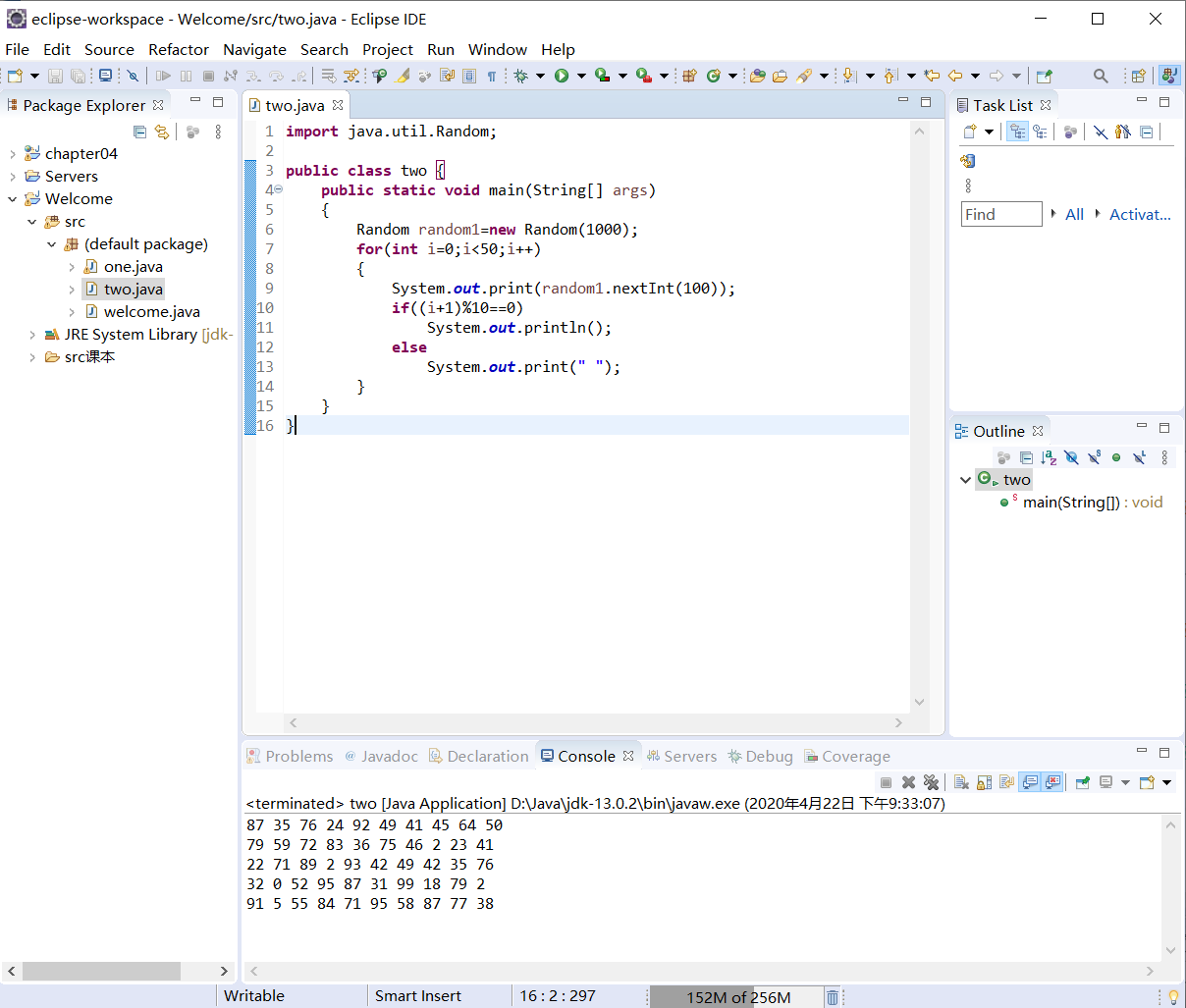
System.***out***.print(" ");

}

}

}

截图：



9.5

代码：

**import** java.util.GregorianCalendar;

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

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

{

GregorianCalendar date=**new** GregorianCalendar();

**int** year=date.get(GregorianCalendar.***YEAR***);

**int** month=date.get(GregorianCalendar.***MONTH***)+1;

**int** day=date.get(GregorianCalendar.***DAY\_OF\_MONTH***);

System.***out***.println(year+"-"+month+"-"+day);

date.setTimeInMillis(1234567898765L);

year=date.get(GregorianCalendar.***YEAR***);

month=date.get(GregorianCalendar.***MONTH***)+1;

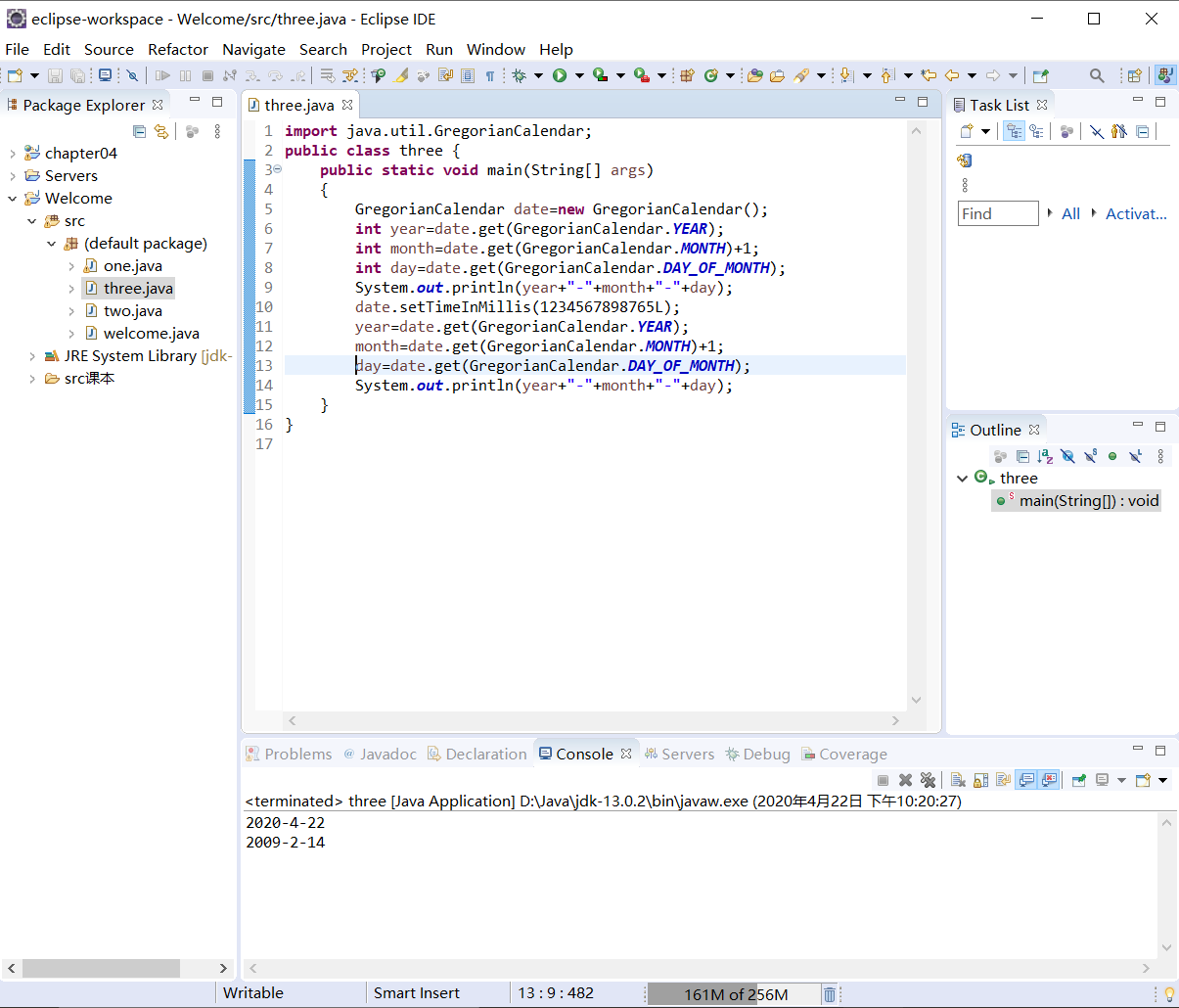
day=date.get(GregorianCalendar.***DAY\_OF\_MONTH***);

System.***out***.println(year+"-"+month+"-"+day);

}

}

截图：



9.9

UML类图：

|  |
| --- |
| RegularPolygon |
| **-**n:int  -side:double  -x:double  -y:double |
| +RegularPolygon()  +RegularPolygon(int,double)  +RegularPolygon(int ,double,double,double)  +getN():int  +setN(int)  +getSide():double  +setSide(double)  +getX():double  +setX(double)  +getY():double  +setY(double)  +getPerimeter():double  +getArea():double |

代码：

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

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

{

RegularPolygon A= **new** RegularPolygon();

RegularPolygon B= **new** RegularPolygon(6,4);

RegularPolygon C= **new** RegularPolygon(10,4,5.6,7.8);

System.***out***.println("the first RegularPolygon's perimeter is "+A.getPerimeter()+",the area is "+A.getArea());

System.***out***.println("the second RegularPolygon's perimeter is "+B.getPerimeter()+",the area is "+B.getArea());

System.***out***.println("the thirst RegularPolygon's perimeter is "+C.getPerimeter()+",the area is "+C.getArea());

}

}

**class** RegularPolygon{

**private** **int** n=3;

**private** **double** side=1;

**private** **double** x=0;

**private** **double** y=0;

**public** RegularPolygon()

{

}

**public** RegularPolygon(**int** nn,**double** s)

{

n=nn;

side=s;

}

**public** RegularPolygon(**int** nn,**double** s, **double** xx,**double** yy)

{

n=nn;

side=s;

x=xx;

y=yy;

}

**public** **int** getN()

{

**return** n;

}

**public** **void** setN(**int** N)

{

n=N;

}

**public** **double** getSide()

{

**return** side;

}

**public** **void** setSide(**double** s)

{

side=s;

}

**public** **double** getX()

{

**return** x;

}

**public** **void** setX(**double** x)

{

**this**.x=x;

}

**public** **double** getY()

{

**return** y;

}

**public** **void** setY(**double** y)

{

**this**.y=y;

}

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

{

**return** n\*side;

}

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

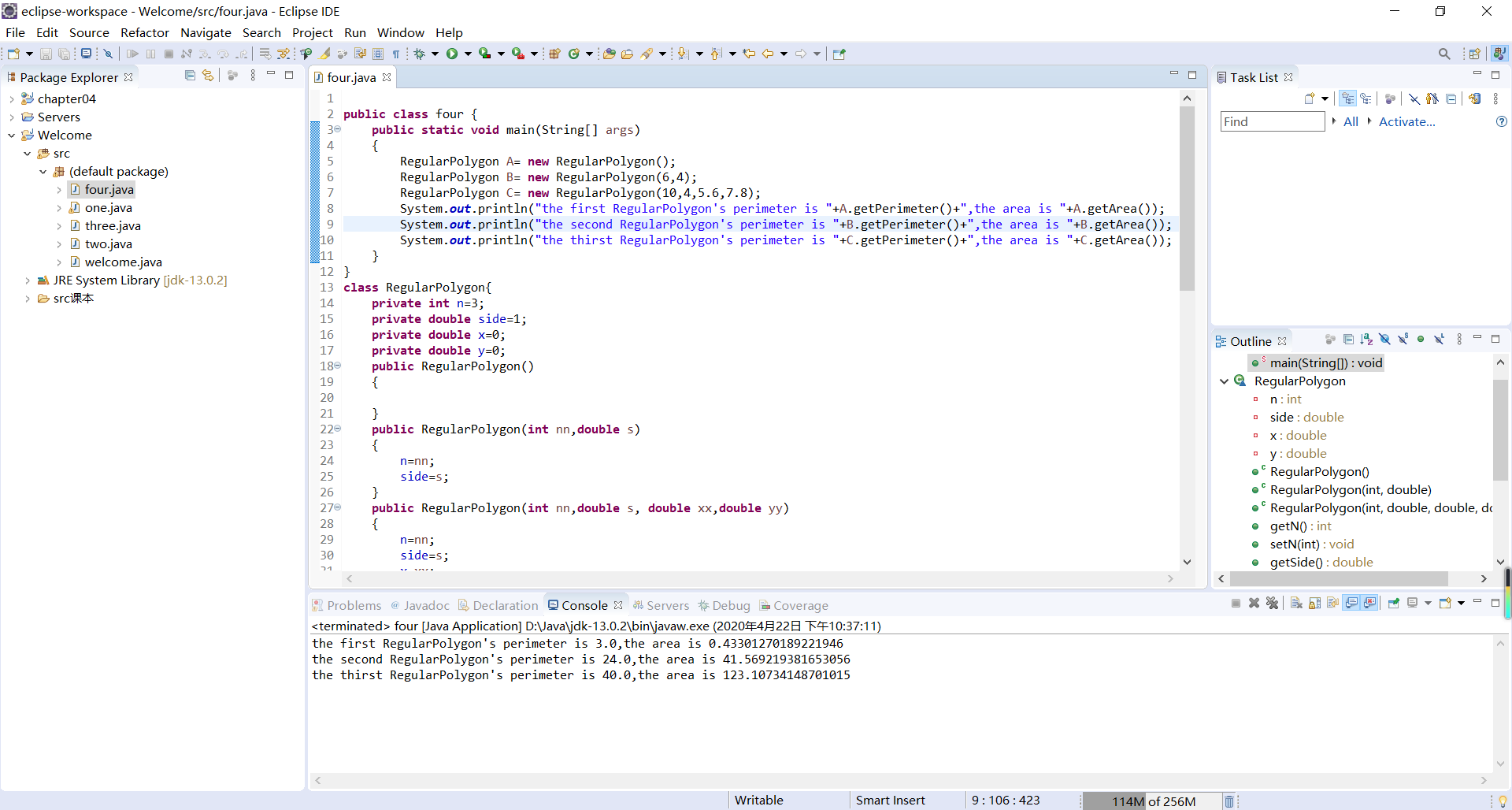
{

**return** (n\*side\*side)/(4\*Math.*tan*(Math.***PI***/n));

}

}

截图：



9.12

代码：

**import** java.util.Scanner;

**public** **class** five {

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

{

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

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

**double** x1=input.nextDouble();

**double** y1=input.nextDouble();

**double** x2=input.nextDouble();

**double** y2=input.nextDouble();

**double** x3=input.nextDouble();

**double** y3=input.nextDouble();

**double** x4=input.nextDouble();

**double** y4=input.nextDouble();

LinearEquation l=**new** LinearEquation(y1-y2,x2-x1,y3-y4,x4-x3,((y1-y2)\*x1-(x1-x2)\*y1),((y3-y4)\*x3-(x3-x4)\*y3));

System.***out***.println("The answer is ( "+l.getX()+" , "+l.getY()+" )");

}

}

**public** **class** LinearEquation {

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

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

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

**private** **double** d;

**private** **double** e;

**private** **double** f;

**public** LinearEquation(**double** A,**double** B,**double** C,**double** D,**double** E,**double** F)

{

a=A;b=B;c=C;d=D;e=E;f=F;

}

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

{

**return** a;

}

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

{

**return** b;

}

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

{

**return** c;

}

**public** **double** getD()

{

**return** d;

}

**public** **double** getE()

{

**return** e;

}

**public** **double** getF()

{

**return** f;

}

**public** **boolean** isSolvable()

{

**if**((a\*d-b\*c)!=0) **return** **true**;

**else** **return** **false**;

}

**public** **double** getX()

{

**double** x=(e\*d-b\*f)/(a\*d-b\*c);

**return** x;

}

**public** **double** getY()

{

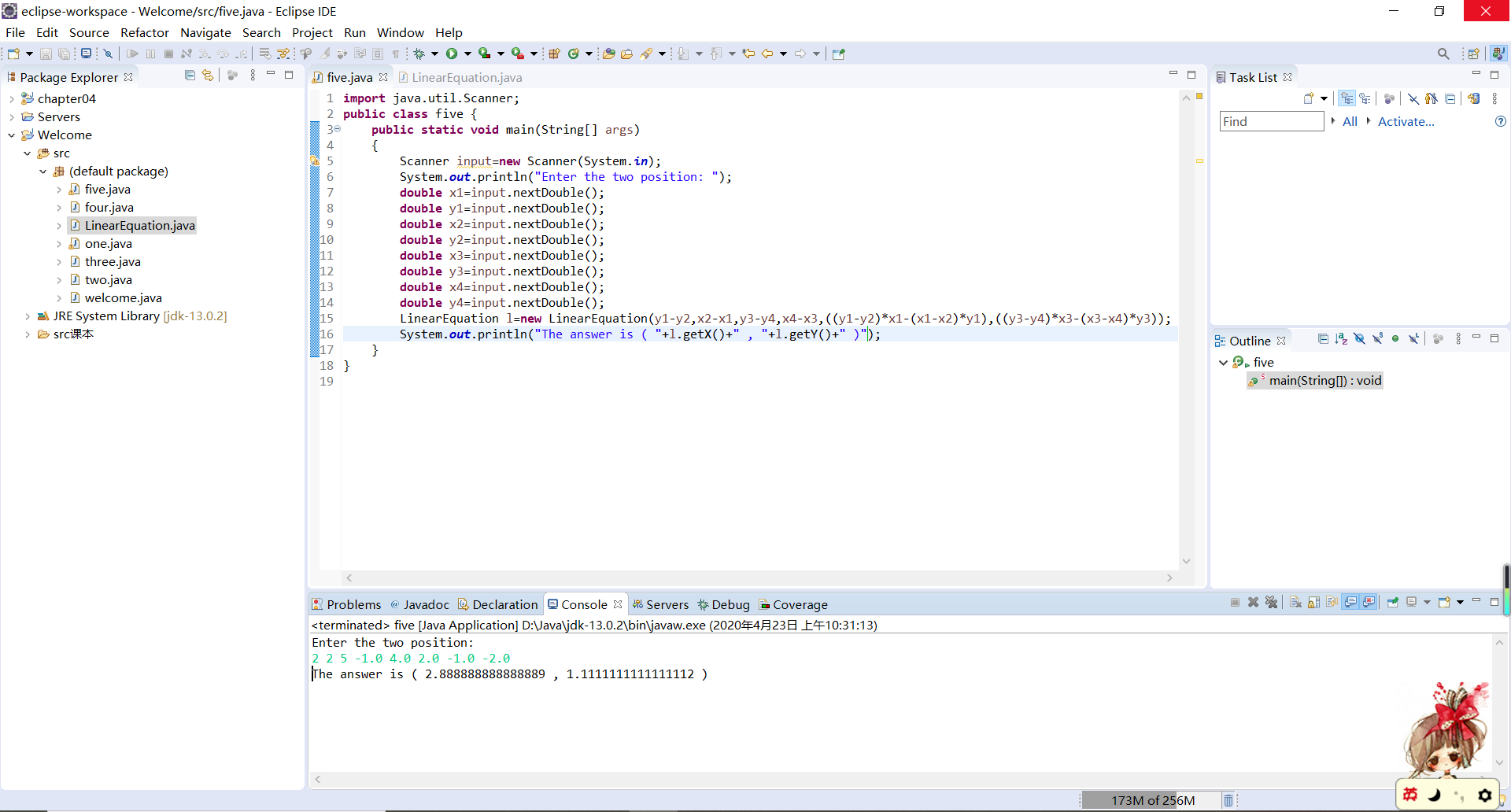
**double** y=(a\*f-e\*c)/(a\*d-b\*c);

**return** y;

}

}

截图：



9.13

代码：

**import** java.util.Scanner;

**public** **class** six {

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

System.***out***.println("Enter the number of rows and columns in the array:");

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

**int** row = input.nextInt();

**int** column = input.nextInt();

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

**double**[][] a = **new** **double**[row][column];

**for** (**int** i = 0; i < row; i++) {

**for** (**int** j = 0; j < column; j++) {

a[i][j] = input.nextDouble();

}

}

System.***out***.println("The location of the largest element is " + *locateLargest*(a).maxValue + " at ( "

+ *locateLargest*(a).row + " , " + *locateLargest*(a).column + " )");

}

**public** **static** Location locateLargest(**double**[][] a) {

Location l = **new** Location();

l.row = 0;

l.column = 0;

l.maxValue = 0;

**for** (**int** i = 0; i < a.length; i++) {

**for** (**int** j = 0; j < a[0].length; j++) {

**if** (a[i][j] > l.maxValue) {

l.row = i;

l.column = j;

l.maxValue = a[i][j];

}

}

}

**return** l;

}

}

**class** Location {

**public** **int** row;

**public** **int** column;

**public** **double** maxValue;

}

截图：

