

Jinan University

Java Programming Lab Report

Major: Computer Science & Technology

School: International School

Student name : _____
(p.s. your name on JNU academic system)

Student number : _____

Date of Submission (mm-dd-yyyy): _____

Instructor: Yuxia Sun

Table of Content

LAB 5	DATE: 4/25/2023	3
Problem 1.	(9.1)	3
Problem 2.	(9.5)	5
Problem 3.	(9.11)	7
Problem 4.	(9.9)(Optional)	11
Problem 5.	(9.13)(Optional)	15

LAB 5 DATE: 4/25/2023

Student Name: _____ Student ID: _____

Problem 1. (9.1)

9.1 (The **Rectangle** class) Following the example of the **Circle** class in Section 9.2, design a class named **Rectangle** to represent a rectangle. The class contains:

- Two **double** data fields named **width** and **height** that specify the width and height of the rectangle. The default values are **1** for both **width** and **height**.
- A no-arg constructor that creates a default rectangle.
- A constructor that creates a rectangle with the specified **width** and **height**.
- A method named **getArea()** that returns the area of this rectangle.
- A method named **getPerimeter()** that returns the perimeter.

Draw the UML diagram for the class then implement the class. Write a test program that creates two **Rectangle** objects—one with width **4** and height **40**, and the other with width **3.5** and height **35.9**. Display the width, height, area, and perimeter of each rectangle in this order.

* Source Code / Solution :

Rectangle
-width:double -height:double
+Rectangle():void +Rectangle(double width, double height):void +getWidth():double +getHeight():double +getArea():double +getPerimeter():double
<pre>public class Rectangle { private final double width; private final double height;</pre>

```
Rectangle(){
    this.width = 1;
    this.height = 1;
}

Rectangle(double width, double height){
    this.width = width;
    this.height = height;
}

public double getWidth(){
    return this.width;
}

public double getHeight(){
    return this.height;
}

public double getArea(){
    return this.height * this.width;
}

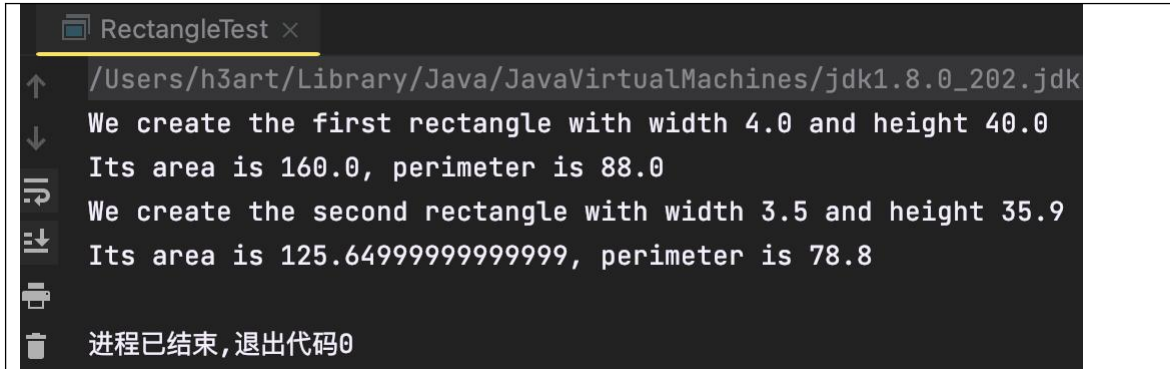
public double getPerimeter(){
    return 2 * (height + width);
}
}

public class RectangleTest {
    public static void main(String[] args) {
        Rectangle rectangle1 = new Rectangle(4, 40);
        System.out.println("We create the first rectangle with width " + rectangle1.getWidth()
            + " and height " + rectangle1.getHeight()
        );
        System.out.println("Its area is " + rectangle1.getArea() + ", perimeter is " +
            rectangle1.getPerimeter());

        Rectangle rectangle2 = new Rectangle(3.5, 35.9);
        System.out.println("We create the second rectangle with width " +
            rectangle2.getWidth()
```

```
        + " and height " + rectangle2.getHeight()
    );
    System.out.println("Its area is " + rectangle2.getArea() + ", perimeter is " +
rectangle2.getPerimeter());
    }
}
```

*** Output:**



```
RectangleTest x
/Users/h3art/Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk
We create the first rectangle with width 4.0 and height 40.0
Its area is 160.0, perimeter is 88.0
We create the second rectangle with width 3.5 and height 35.9
Its area is 125.64999999999999, perimeter is 78.8
进程已结束,退出代码0
```

*** Debugging/Testing:**

Bug1: In the construction method, if the incoming parameter has the same name as the class variable, using such as width = width cannot initialize the class variable.

Fix: I can use "this" to modify class variables to achieve the effect of assigning variables with the same name to class variables.

Problem 2. (9.5)

- *9.5** (Use the **GregorianCalendar** class) Java API has the **GregorianCalendar** class in the **java.util** package, which you can use to obtain the year, month, and day of a date. The no-arg constructor constructs an instance for the current date, and the methods **get (GregorianCalendar.YEAR)**, **get (GregorianCalendar.MONTH)**, and **get (GregorianCalendar.DAY_OF_MONTH)** return the year, month, and day. Write a program to perform two tasks:
1. Display the current year, month, and day.
 2. The **GregorianCalendar** class has the **setTimeInMillis(long)**, which can be used to set a specified elapsed time since January 1, 1970. Set the value to **1234567898765L** and display the year, month, and day.

*** Source Code / Solution :**

```
import java.util.GregorianCalendar;

public class GregorianCalendarTest {
```

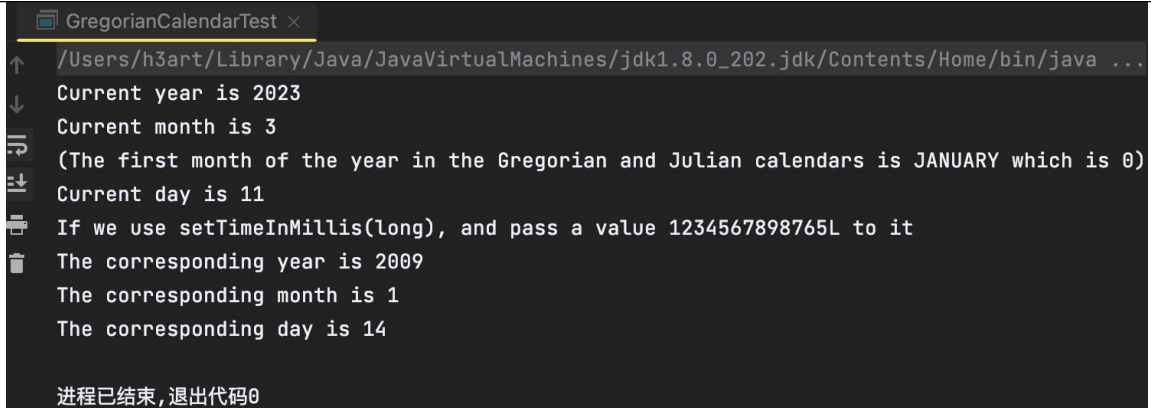
```
public static void main(String[] args) {
    GregorianCalendar calendar = new GregorianCalendar();

    System.out.println("Current year is " + calendar.get(GregorianCalendar.YEAR));
    // The first month of the year in the Gregorian and Julian calendars is JANUARY which is
    0
    System.out.println("Current month is " + calendar.get(GregorianCalendar.MONTH));
    System.out.println("(The first month of the year in the Gregorian and Julian calendars
    is JANUARY which is 0)");
    System.out.println("Current day is " +
    calendar.get(GregorianCalendar.DAY_OF_MONTH));

    // Method setTimeInMillis(long) can be used to set a specified elapsed time since
    January 1, 1970
    calendar.setTimeInMillis(1234567898765L);
    System.out.println("If we use setTimeInMillis(long), and pass a value
    1234567898765L to it");

    System.out.println("The corresponding year is " +
    calendar.get(GregorianCalendar.YEAR));
    System.out.println("The corresponding month is " +
    calendar.get(GregorianCalendar.MONTH));
    System.out.println("The corresponding day is " +
    calendar.get(GregorianCalendar.DAY_OF_MONTH));
}
```

*** Output:**



```
GregorianCalendarTest x
/Users/h3art/Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk/Contents/Home/bin/java ...
Current year is 2023
Current month is 3
(The first month of the year in the Gregorian and Julian calendars is JANUARY which is 0)
Current day is 11
If we use setTimeInMillis(long), and pass a value 1234567898765L to it
The corresponding year is 2009
The corresponding month is 1
The corresponding day is 14
进程已结束,退出代码0
```

*** Debugging/Testing:**

Bug1: I thought the program was outputting the wrong month, it always return a month number which has difference 1 to current month.

Fix: Through reading the corresponding method document, I found the first month of the year in the Gregorian and Julian calendars is JANUARY which is 0.

Problem 3. (9.11)

***9.11** (Algebra: 2×2 linear equations) Design a class named **LinearEquation** for a 2×2 system of linear equations:

$$\begin{array}{l} ax + by = e \\ cx + dy = f \end{array} \quad x = \frac{ed - bf}{ad - bc} \quad y = \frac{af - ec}{ad - bc}$$

The class contains:

- Private data fields **a**, **b**, **c**, **d**, **e**, and **f**.
- A constructor with the arguments for **a**, **b**, **c**, **d**, **e**, and **f**.
- Six getter methods for **a**, **b**, **c**, **d**, **e**, and **f**.
- A method named **isSolvable()** that returns true if $ad - bc$ is not 0.
- Methods **getX()** and **getY()** that return the solution for the equation.

Draw the UML diagram for the class then implement the class. Write a test program that prompts the user to enter **a**, **b**, **c**, **d**, **e**, and **f** and displays the result. If $ad - bc$ is 0, report that “The equation has no solution.” See Programming Exercise 3.3 for sample runs.

*** Source Code / Solution :**

LinearEquation
-a:double -b:double -c:double -d:double -e:double -f:double
+LinearEquation(double a, double b, double c, double d, double e, double f):void +getA():double +getB():double +getC():double +getD():double +getE():double +getF():double +isSolvable():boolean +getX():double +getY():double
<pre> public class LinearEquation { /** * $ax + by = e$ * $cx + dy = f$ * $x = (ed - bf) / (ad - bc)$ * $y = (af - ec) / (ad - bc)$ */ private final double a; private final double b; private final double c; private final double d; private final double e; private final double f; LinearEquation(double a, double b, double c, double d, double e, double f) { this.a = a; this.b = b; this.c = c; this.d = d; this.e = e; this.f = f; } </pre>


```
public double getA() {  
    return this.a;  
}  
  
public double getB() {  
    return this.b;  
}  
  
public double getC() {  
    return this.c;  
}  
  
public double getD() {  
    return this.d;  
}  
  
public double getE() {  
    return this.e;  
}  
  
public double getF() {  
    return this.f;  
}  
  
public boolean isSolvable() {  
    return a * d - b * c != 0;  
}  
  
public double getX() {  
    return (e * d - b * f) / (a * d - b * c);  
}  
  
public double getY() {  
    return (a * f - e * c) / (a * d - b * c);  
}  
}  
  
import java.util.Scanner;
```

```
public class LinearEquationTest {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        double a, b, c, d, e, f;

        System.out.println("Suppose we have a 2 * 2 system of linear equations: ");
        System.out.println("\tax + by = e\n" + "\tcx + dy = f");
        System.out.println("This program will help you judge whether it can be solve,");
        System.out.println("if can, it will output the corresponding x and y");

        System.out.print("Enter the factors(Order: a, b, c, d, e, f): ");
        a = input.nextDouble();
        b = input.nextDouble();
        c = input.nextDouble();
        d = input.nextDouble();
        e = input.nextDouble();
        f = input.nextDouble();

        LinearEquation linearEquation = new LinearEquation(a, b, c, d, e, f);

        if(!linearEquation.isSolvable()){
            System.out.println("The equation has no solution.");
        }else{
            System.out.println("The solution x is " + linearEquation.getX());
            System.out.println("The solution y is " + linearEquation.getY());
        }
    }
}
```

*** Output:**

```
LinearEquationTest x
/Users/h3art/Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk
Suppose we have a 2 * 2 system of linear equations:
    ax + by = e
    cx + dy = f
This program will help you judge whether it can be solve,
if can, it will output the corresponding x and y
Enter the factors(Order: a, b, c, d, e, f): 1 2 3 4 5 6
The solution x is -4.0
The solution y is 4.5

进程已结束,退出代码0

LinearEquationTest x
/Users/h3art/Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk
Suppose we have a 2 * 2 system of linear equations:
    ax + by = e
    cx + dy = f
This program will help you judge whether it can be solve,
if can, it will output the corresponding x and y
Enter the factors(Order: a, b, c, d, e, f): 1 2 2 4 5 6
The equation has no solution.

进程已结束,退出代码0
```

*** Debugging/Testing:**

Bug1: Using the wrong type (int) to store the variable value, resulting in the wrong answer when calculating the division.

Fix: Modify all the class variables of the linearEquation to double type.

Problem 4. (9.9)(Optional)

- **9.9** (Geometry: *n*-sided regular polygon) In an *n*-sided regular polygon, all sides have the same length and all angles have the same degree (i.e., the polygon is both equilateral and equiangular). Design a class named **RegularPolygon** that contains:
- A private **int** data field named **n** that defines the number of sides in the polygon with default value **3**.

- A private **double** data field named **side** that stores the length of the side with default value **1**.
- A private **double** data field named **x** that defines the x-coordinate of the polygon's center with default value **0**.
- A private **double** data field named **y** that defines the y-coordinate of the polygon's center with default value **0**.
- A no-arg constructor that creates a regular polygon with default values.
- A constructor that creates a regular polygon with the specified number of sides and length of side, centered at **(0, 0)**.
- A constructor that creates a regular polygon with the specified number of sides, length of side, and x- and y-coordinates.
- The accessor and mutator methods for all data fields.
- The method **getPerimeter()** that returns the perimeter of the polygon.
- The method **getArea()** that returns the area of the polygon. The formula for computing the area of a regular polygon is

$$\text{Area} = \frac{n \times s^2}{4 \times \tan\left(\frac{\pi}{n}\right)}.$$

Draw the UML diagram for the class then implement the class. Write a test program that creates three **RegularPolygon** objects, created using the no-arg constructor, using **RegularPolygon(6, 4)**, and using **RegularPolygon(10, 4, 5.6, 7.8)**. For each object, display its perimeter and area.

*** Source Code / Solution :**

RegularPolygon
-n:int -side:double -x:double -y:double
+RegularPolygon():void +RegularPolygon(int numOfSide, double lenOfSide):void +RegularPolygon(int numOfSide, double lenOfSide, double centerX, double centerY):void +getN():int +setN():void +getSide():double +setSide():void +getX():double +setX():void +getY():double +setY():void +getPerimeter():double +getArea():double

```
public class RegularPolygon {
    private int n; // number of side
    private double side; // length of side
    private double x; // x-coordinate of the polygon's center
    private double y; // y-coordinate of the polygon's center

    RegularPolygon() {
        this.n = 3;
        this.side = 1;
        this.x = 0;
        this.y = 0;
    }

    RegularPolygon(int numOfSide, double lenOfSide) {
        this.n = numOfSide;
        this.side = lenOfSide;
        this.x = 0;
        this.y = 0;
    }

    RegularPolygon(int numOfSide, double lenOfSide, double centerX, double centerY) {
        this.n = numOfSide;
        this.side = lenOfSide;
        this.x = centerX;
        this.y = centerY;
    }

    public int getN() {
        return this.n;
    }

    public void setN(int n) {
        this.n = n;
    }

    public double getSide() {
        return this.side;
    }
}
```

```
public void setSide(double side) {
    this.side = side;
}

public double getX() {
    return this.x;
}

public void setX(double x) {
    this.x = x;
}

public double getY() {
    return this.y;
}

public void setY(double y) {
    this.y = y;
}

public double getPerimeter() {
    return this.n * this.side;
}

public double getArea() {
    return n * Math.pow(this.side, 2) / 4 * Math.tan(Math.PI / this.n);
}
}

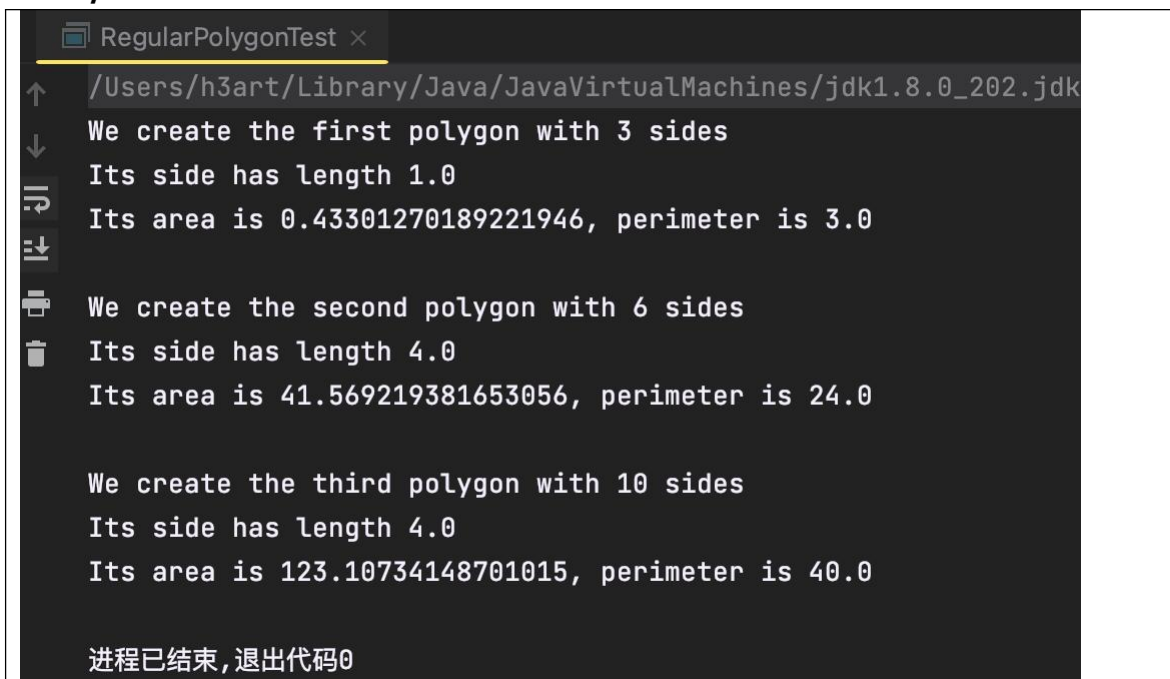
public class RegularPolygonTest {
    public static void main(String[] args) {
        RegularPolygon regularPolygon1 = new RegularPolygon();
        System.out.println("We create the first polygon with " + regularPolygon1.getN() + "
sides");
        System.out.println("Its side has length " + regularPolygon1.getSide());
        System.out.println("Its area is " + regularPolygon1.getArea() + ", perimeter is "
+ regularPolygon1.getPerimeter());
        System.out.println();

        RegularPolygon regularPolygon2 = new RegularPolygon(6, 4);
```

```
System.out.println("We create the second polygon with " + regularPolygon2.getN() +
"sides");
System.out.println("Its side has length " + regularPolygon2.getSide());
System.out.println("Its area is " + regularPolygon2.getArea() + ", perimeter is "
+ regularPolygon2.getPerimeter());
System.out.println();

RegularPolygon regularPolygon3 = new RegularPolygon(10, 4, 5.6, 7.8);
System.out.println("We create the third polygon with " + regularPolygon3.getN() +
"sides");
System.out.println("Its side has length " + regularPolygon3.getSide());
System.out.println("Its area is " + regularPolygon3.getArea() + ", perimeter is "
+ regularPolygon3.getPerimeter());
}
}
```

*** Output:**



```
RegularPolygonTest x
/Users/h3art/Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk
We create the first polygon with 3 sides
Its side has length 1.0
Its area is 0.43301270189221946, perimeter is 3.0

We create the second polygon with 6 sides
Its side has length 4.0
Its area is 41.569219381653056, perimeter is 24.0

We create the third polygon with 10 sides
Its side has length 4.0
Its area is 123.10734148701015, perimeter is 40.0

进程已结束,退出代码0
```

*** Debugging/Testing:**

Bug1: Parentheses are omitted in the continuous formula when calculating area, and the operation priority is wrong.
Fix: Add parentheses to the two expressions on both sides of the division sign.

Problem 5. (9.13)(Optional)

****9.13** (The **Location** class) Design a class named **Location** for locating a maximal value and its location in a two-dimensional array. The class contains public data fields **row**, **column**, and **maxValue** that store the maximal value and its indices in a two-dimensional array with **row** and **column** as **int** types and **maxValue** as a **double** type.

Write the following method that returns the location of the largest element in a two-dimensional array:

```
public static Location locateLargest(double[][] a)
```

The return value is an instance of **Location**. Write a test program that prompts the user to enter a two-dimensional array and displays the location of the largest element in the array. Here is a sample run:

```
Enter the number of rows and columns in the array: 3 4 
Enter the array:
23.5 35 2 10 
4.5 3 45 3.5 
35 44 5.5 9.6 
The location of the largest element is 45 at (1, 2)
```

*** Source Code / Solution :**

```
public class Location {
    public int row;
    public int column;
    public double maxValue;

    Location(){
        this.row = 0;
        this.column = 0;
        this.maxValue = Double.MIN_VALUE;
    }
}

import java.util.Scanner;

public class LocationTest {
    public static Location locateLargest(double[][] a) {
        Location result = new Location();
        int lenOfRow = a.length;
        int lenOfColumn = a[0].length;
```



```
    for (int i = 0; i < lenOfRow; i++) {
        for (int j = 0; j < lenOfColumn; j++) {
            if (a[i][j] > result.maxValue) {
                result.maxValue = a[i][j];
                result.row = i;
                result.column = j;
            }
        }
    }
    return result;
}

public static void main(String[] args) {
    Scanner input = new Scanner(System.in);

    System.out.print("Enter the number of rows and columns in the array: ");
    int numberOfRow = input.nextInt();
    int numberOfColumn = input.nextInt();

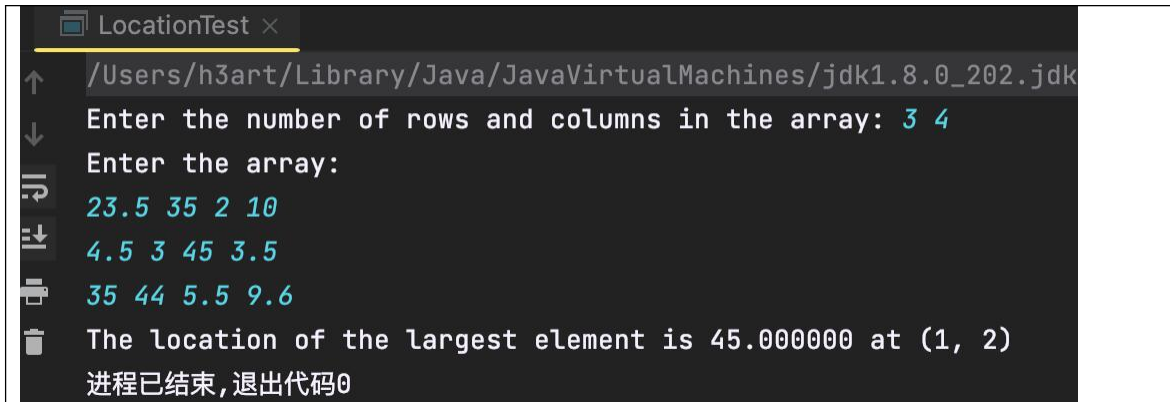
    double[][] testArray = new double[numberOfRow][numberOfColumn];

    System.out.println("Enter the array:");
    for (int i = 0; i < numberOfRow; i++) {
        for (int j = 0; j < numberOfColumn; j++) {
            testArray[i][j] = input.nextDouble();
        }
    }

    Location findResult = locateLargest(testArray);

    System.out.printf("The location of the largest element is %f at (%d, %d)",
        findResult.maxValue, findResult.row, findResult.column);
}
}
```

*** Output:**



```
LocationTest x
/Users/h3art/Library/Java/JavaVirtualMachines/jdk1.8.0_202.jdk
Enter the number of rows and columns in the array: 3 4
Enter the array:
23.5 35 2 10
4.5 3 45 3.5
35 44 5.5 9.6
The location of the largest element is 45.000000 at (1, 2)
进程已结束,退出代码0
```

*** Debugging/Testing:**

Bug1: Array index processing error.

Fix: Change the index processing range from 0 to the number of row - 1/the number of column - 1.