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 3 DATE: 3/28/2023

Student Name:_____ **Student ID:** _____

Problem 1. (5.9)

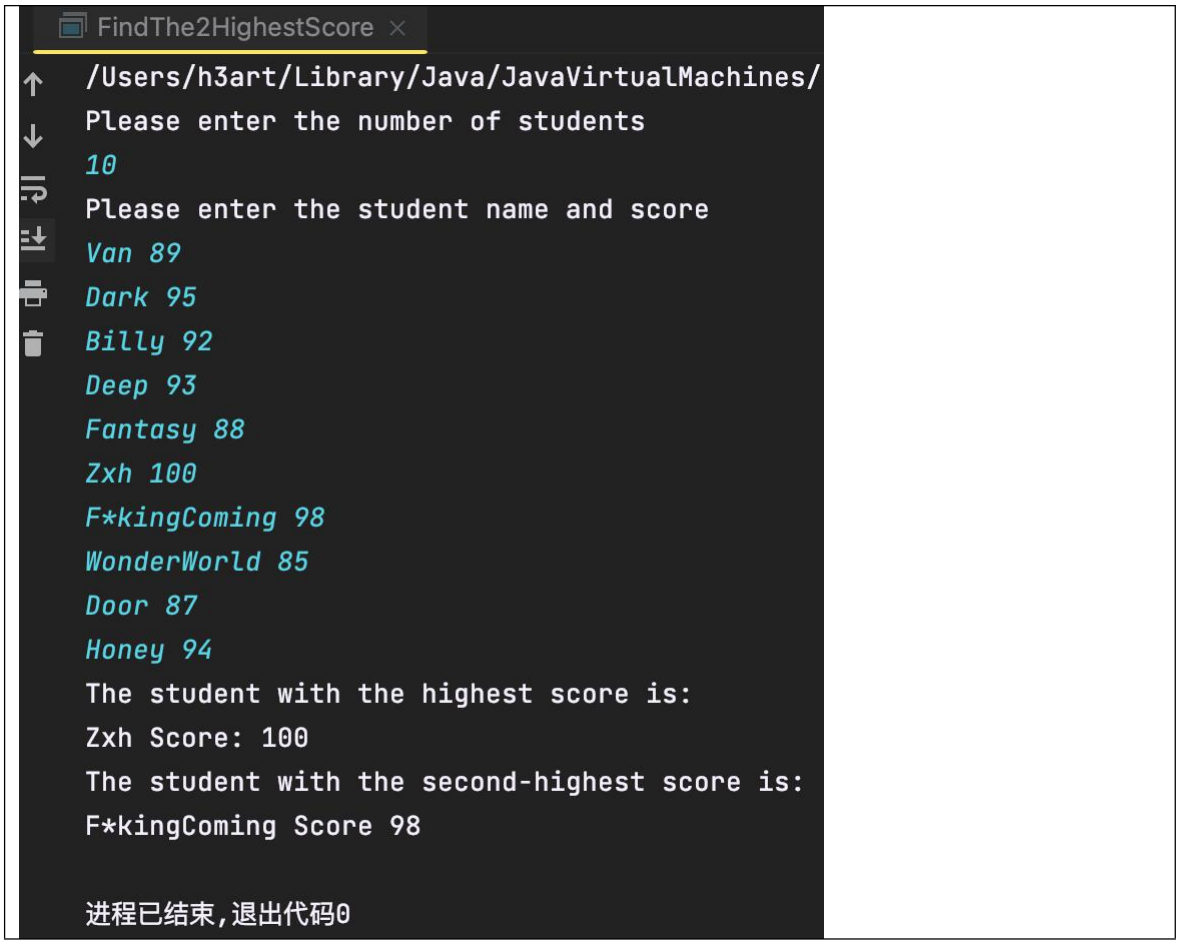
***5.9** (*Find the two highest scores*) Write a program that prompts the user to enter the number of students and each student's name and score, and finally displays the student with the highest score and the student with the second-highest score. Use the `next()` method in the `Scanner` class to read a name rather than using the `nextLine()` method. Assume that the number of students is at least 2.

**** Source Code / Solution :***

```
import java.util.Scanner;

public class FindThe2HighestScore {

   public static void main(String[] args) {
      int studentNumber;
      int topScore = -1, secondScore = -1;
      String topStudent = "", secondStudent = "";

      Scanner input = new Scanner(System.in);

      System.out.println("Please enter the number of students");
      studentNumber = input.nextInt();
      input.close();

      if(studentNumber<2){
         throw new IllegalArgumentException("The number of students is at least 2");
      }

      System.out.println("Please enter the student name and score");
      for (int i = 0; i < studentNumber; i++) {
         String name = input.next();
         int score = input.nextInt();

         if(score > topScore){
            secondScore = topScore;
```

```
            secondStudent = topStudent;
            topScore = score;
            topStudent = name;
          }else if(score > secondScore){
            secondScore = score;
            secondStudent = name;
          }
        }

        //displays the student with the highest score and the student with the second-highest
score
        System.out.println("The student with the highest score is:");
        System.out.println(topStudent + " Score: " + topScore);

        System.out.println("The student with the second-highest score is:");
        System.out.println(secondStudent + " Score " + secondScore);

    }
}
```

**** Output:***



**** Debugging/Testing:***

Bug1: The algorithm I designed to find the second highest grade is wrong, can't find the second highest grade.
Fix: The variable assignment in if-else was redesigned.

Problem 2. (5.29)(Optional)

5.29 (*Display calendars*) Write a program that prompts the user to enter the year and first day of the year and displays the calendar table for the year on the console. For example, if the user entered the year **2013, and **2** for Tuesday, January 1, 2013, your program should display the calendar for each month in the year, as follows:

January 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

. . .

December 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

**** Source Code / Solution :***

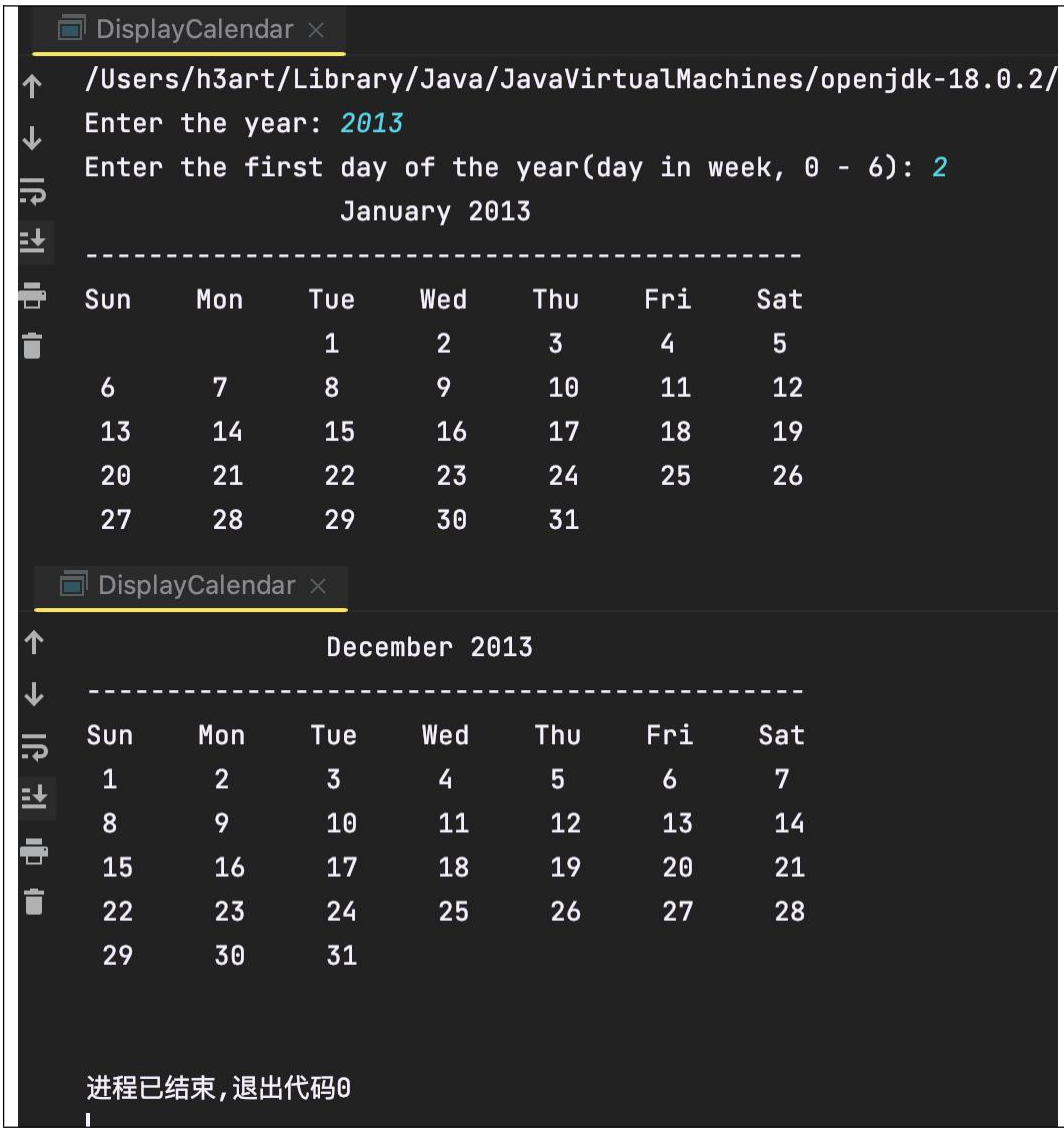
```
import java.util.Scanner;

public class DisplayCalendar {
   final static String[] Months = {
         "January",
         "February",
         "March",
         "April",
         "May",
         "June",
         "July",
         "August",
         "September",
         "October",
         "November",
```

```
        "December"
    };

    private static boolean isLeapYear(int year) {
        if (year % 400 == 0) {
            return true;
        } else return (year % 4 == 0) && (year % 100 != 0);
    }

    private static int getDaysInMonth(int year, int month) {
        int daysInMonth;
        switch (month) {
            case 2:
                if (isLeapYear(year)) {
                    daysInMonth = 29;
                } else {
                    daysInMonth = 28;
                }
                break;
            case 4:
            case 6:
            case 9:
            case 11:
                daysInMonth = 30;
                break;
            default:
                daysInMonth = 31;
        }
        return daysInMonth;
    }

    private static void printCalendar(int year, int firstDay) {
        int dayOfWeek = firstDay;

        for (int month = 1; month <= 12; month++) {
            System.out.printf("%28s\n", Months[month - 1] + ' ' + year);
            System.out.println("-----------------------------------------");
            System.out.println("Sun  Mon  Tue  Wed  Thu  Fri  Sat");
```

```
        if (dayOfWeek != 7) {
            for (int gap = dayOfWeek; gap > 0; gap--) {
                System.out.print("       ");
            }
            System.out.print(" ");
        } else {
            dayOfWeek = 0;
        }

        int dayLimit = getDaysInMonth(year, month);
        for (int day = 1; day <= dayLimit; day++, dayOfWeek++) {
            if (dayOfWeek == 7) {
                dayOfWeek = 0;
                System.out.print("\n");
            }

            if (dayOfWeek == 0) {
                System.out.print(" ");
            }
            System.out.printf("%-7d", day);
        }

        System.out.println("\n");
    }
}


public static void main(String[] args) {
    int year, firstDay;

    Scanner input = new Scanner(System.in);

    System.out.print("Enter the year: ");
    year = input.nextInt();
    System.out.print("Enter the first day of the year(day in week, 0 - 6): ");
    firstDay = input.nextInt();
    input.close();

    printCalendar(year, firstDay);
```

```
    }
}
```

**** Output:***



**** Debugging/Testing:***

Bug1: Incorrect setting of Spaces between calendar elements, resulting in formatting errors.(the program works

```
properly)
Fix: Keep debugging to find formatting errors and fix them.
```

Problem 3. (6.5)

***6.5** (*Sort three numbers*) Write a method with the following header to display three numbers in increasing order:

```
public static void displaySortedNumbers(
  double num1, double num2, double num3)
```

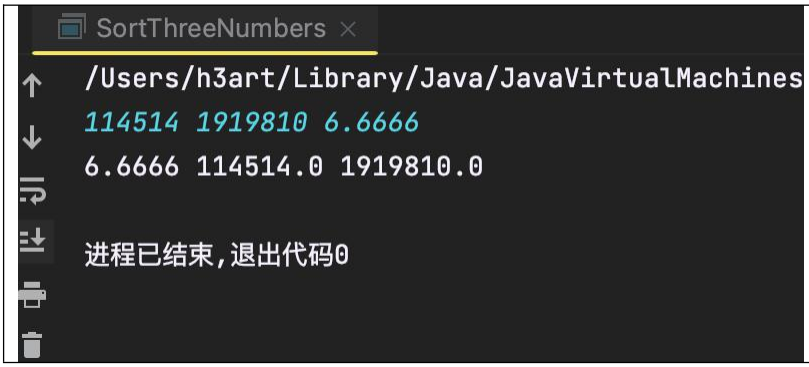
Write a test program that prompts the user to enter three numbers and invokes the method to display them in increasing order.

**** Source Code / Solution :***

```
import java.util.Scanner;

public class SortThreeNumbers {
   final static int LEN = 3;

   public static void displaySortedNumbers(double num1, double num2, double num3){
      if(num1 > num2){
         if(num1 > num3){
            if(num2 > num3){
               System.out.println(num3 + " " + num2 + " " + num1);
            }else{
               System.out.println(num2 + " " + num3 + " " + num1);
            }
         }else{
            System.out.println(num2 + " " + num1 + " " + num3);
         }
      }else{
         if(num2 > num3){
            if(num1 > num3){
               System.out.println(num3 + " " + num1 + " " + num2);
            }else{
               System.out.println(num1 + " " + num3 + " " + num2);
            }
         }else{
            System.out.println(num1 + " " + num2 + " " + num3);
         }
      }
```

```
    }

    public static void main(String[] args) {
        double[] numbers = new double[LEN];

        Scanner input = new Scanner(System.in);

        for(int i = 0; i < LEN; i++){
            numbers[i] = input.nextDouble();
        }

        displaySortedNumbers(numbers[0], numbers[1], numbers[2]);
    }
}
```

**** Output:***



**** Debugging/Testing:***

Bug1: Multiple if-else statements cause an error in sorting management.
Fix: Test and debug the program in each case.

Problem 4. (6.11)

6.11 (*Financial application: compute commissions*) Write a method that computes the commission, using the scheme in Programming Exercise 5.39. The header of the method is as follows:

```
public static double computeCommission(double salesAmount)
```

Write a test program that displays the following table:

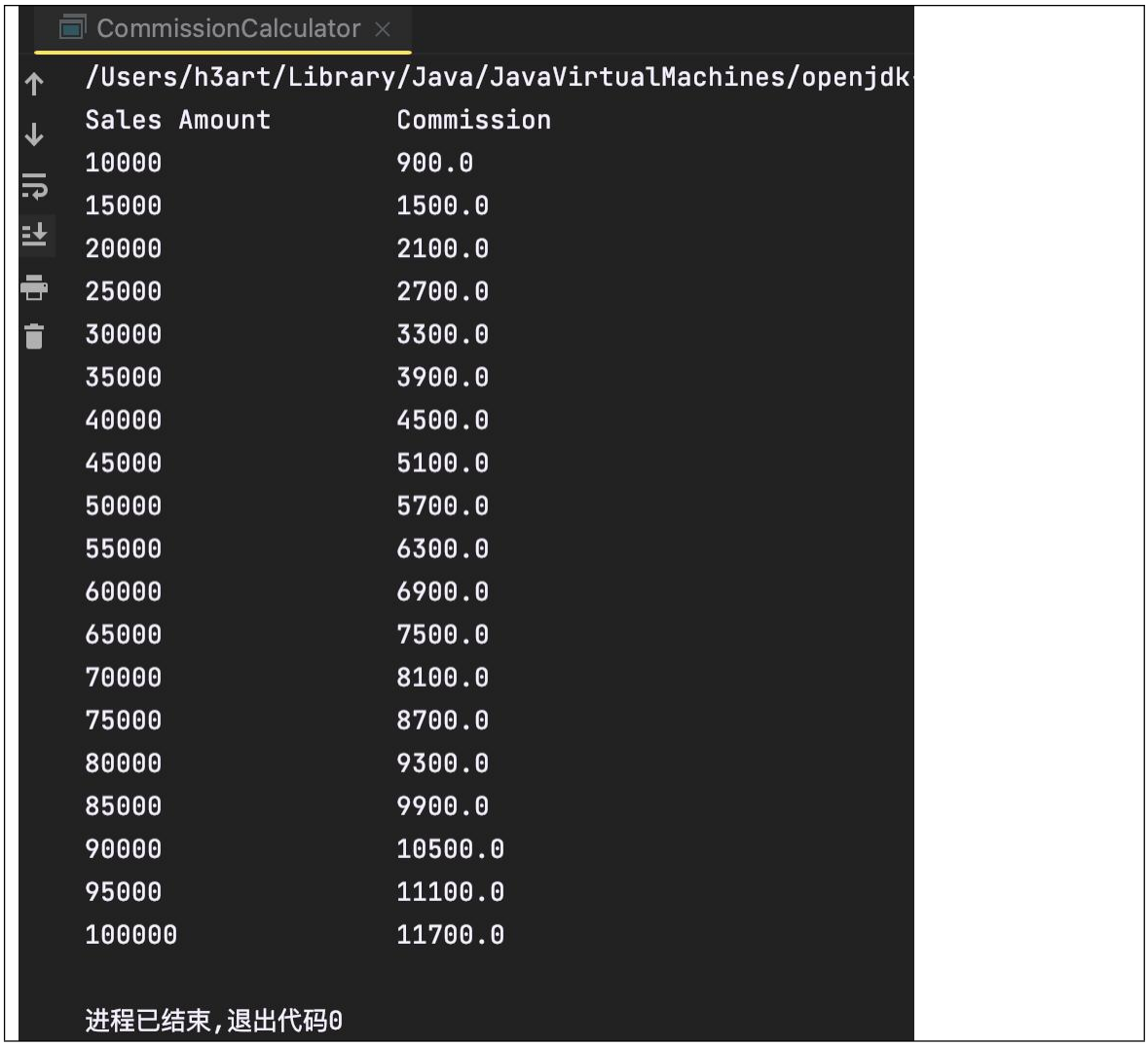
```
Sales Amount        Commission
10000               900.0
15000               1500.0
20000               2100.0
25000               2700.0
30000               3300.0
35000               3900.0
40000               4500.0
45000               5100.0
50000               5700.0
55000               6300.0
60000               6900.0
65000               7500.0
70000               8100.0
75000               8700.0
80000               9300.0
85000               9900.0
90000               10500.0
95000               11100.0
100000              11700.0
```

**** Source Code / Solution :***

```
public class CommissionCalculator {
   public static double computeCommission(double salesAmount){
      double commission;
      if(salesAmount > 10000){
         commission = (salesAmount - 10000) * 0.12 + 900;
      }else if(salesAmount > 5000){
         commission = (salesAmount - 5000) * 0.1 + 400;
      }else {
         commission = salesAmount * 0.08;
      }
```

```
        return commission;
    }

    public static void displayTable(int minAmount, int maxAmount, int offset){
        System.out.println("Sales Amount\t\tCommission");
        for(int amount = minAmount; amount <= maxAmount; amount += offset){
            System.out.println(amount + "\t\t\t\t" + computeCommission(amount));
        }
    }

    public static void main(String[] args) {
        displayTable(10000, 100000, 5000);
    }
}
```

**** Output:***



**** Debugging/Testing:***

Bug1: The condition of the if-else statement is incorrectly set. ⩾ is used as >.
Fix: Debug and fix conditions in if-else statement.

Problem 5. (6.19)(Optional)

***6.19** (*Triangles*) Implement the following two methods:

```
/** Return true if the sum of every two sides is
 * greater than the third side. */
public static boolean isValid(
  double side1, double side2, double side3)

/** Return the area of the triangle. */
public static double area(
  double side1, double side2, double side3)
```

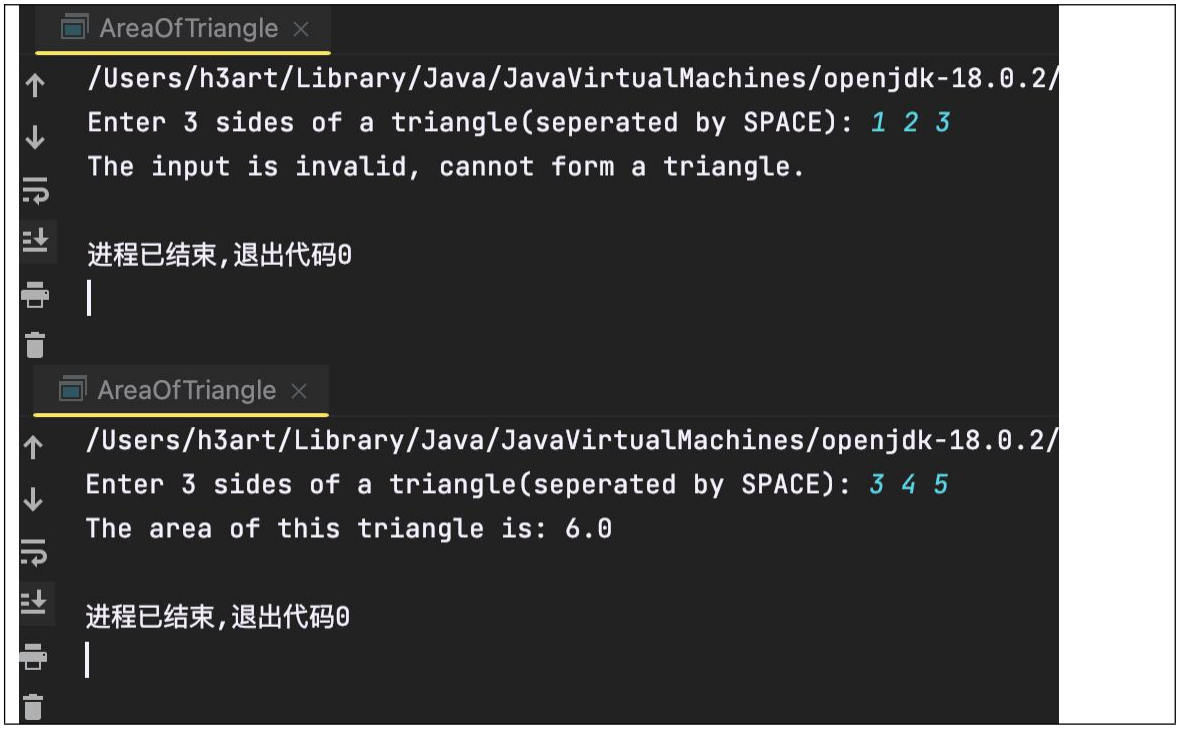
Write a test program that reads three sides for a triangle and uses the **isValid** method to test if the input is valid and uses the **area** method to obtain the area. The program displays the area if the input is valid. Otherwise, it displays that the input is invalid. The formula for computing the area of a triangle is given in Programming Exercise 2.19.

**** Source Code / Solution :***

```
import java.util.Scanner;

public class AreaOfTriangle {
  final static int SIDENUM = 3;

  /* Return true if the sum of every two sides is greater than the third side. */
  public static boolean isValid(double side1, double side2, double side3) {
    return side1 + side2 > side3 &&
        side2 + side3 > side1 &&
        side1 + side3 > side2;
  }

  /* Return the area of the triangle. */
  public static double area(double side1, double side2, double side3) {
    double sFactor = (side1 + side2 + side3) / 2;
    return Math.sqrt(sFactor * (sFactor - side1) * (sFactor - side2) * (sFactor - side3));
  }

  public static void main(String[] args) {
    double[] sides = new double[SIDENUM];

    Scanner input = new Scanner(System.in);

    System.out.print("Enter 3 sides of a triangle(seperated by SPACE): ");
    for (int i = 0; i < SIDENUM; i++) {
```

```
            sides[i] = input.nextDouble();
        }

        if (isValid(sides[0], sides[1], sides[2])) {
            System.out.println("The area of this triangle is: " + area(sides[0], sides[1], sides[2]));
        } else {
            System.out.println("The input is invalid, cannot form a triangle.");
        }

    }
}
```

**** Output:***



**** Debugging/Testing:***

Bug1: The operator priority is omitted in the area calculation, which results in a wrong area output.
Fix: I need to pay attention to operator priority and the use of parentheses in long expressions.