Object-Oriented Programming

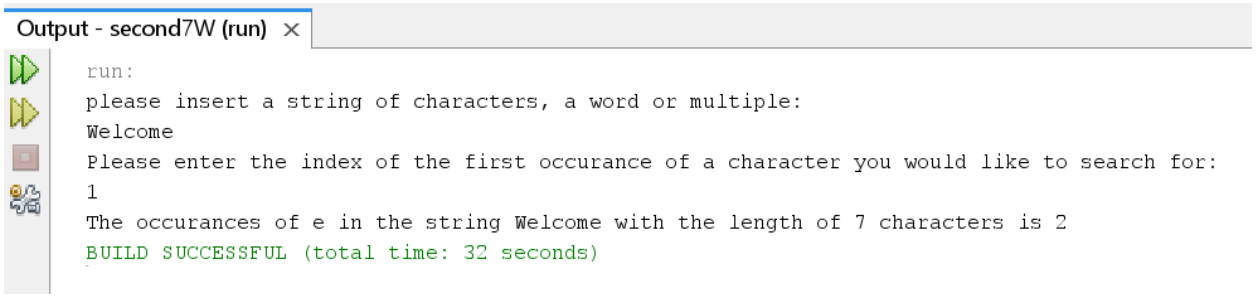
The assignments are Programming Exercises 6.23 on pages 241-242, 7.9 on page 280, 8.1 on pages 307-308, 9.1 on page 362, and 11.1 on page 447 of the textbook (Chapters 6-9, 11). For your convenience, the exercises are shown below, where 9.1 and 11.1 are modified slightly. Please copy here your source codes, including your input and output screenshots. Please upload this document along with your source files (i.e., the .java files) on Blackboard by the due date.

6.23 (*Occurrences of a specified character*) Write a method that finds the number of occurrences of a specified character in a string using the following header:

`public static int count (String str, char a)`

For example, count (“Welcome”, ‘e’) returns 2. Write a test program that prompts the user to enter a string followed by a character then displays the number of occurrences of the character in the string.

```
import java.util.Scanner;

public class HW4number1 {

    public static void main(String[] args) {

        Scanner input = new Scanner(System.in);

        System.out.println("please insert a string of characters, a word or multiple: ");

        String str = input.nextLine();

        System.out.println("Please enter the index of the first occurance of a character " +

                "you would like to search for: ");

        int x = input.nextInt();

        char a = str.charAt(x);

        System.out.println("The occurances of " + a + " in the string " + str +

                " with the length of " + str.length() + " characters is " + count(str,a));

    }

    public static int count(String str, char a) {

        int occTotal = 0;

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

            if (str.charAt(i) == a){

                occTotal++;
            }
        }
        return occTotal;
    }
}
```


Output - second7W (run)

```
run:
please insert a string of characters, a word or multiple:
Welcome
Please enter the index of the first occurance of a character you would like to search for:
1
The occurances of e in the string Welcome with the length of 7 characters is 2
BUILD SUCCESSFUL (total time: 32 seconds)
```

7.9 (*Find the smallest element*) Write a method that finds the smallest element in an array of double values using the following header:

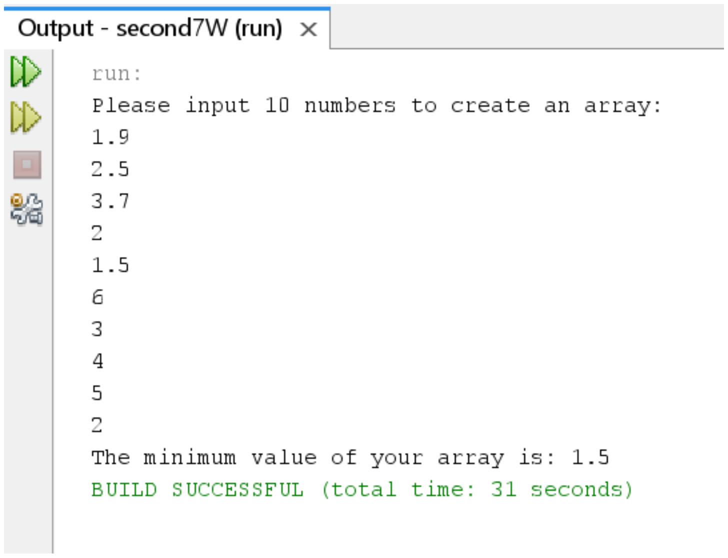
public static double min (double [] array)

Write a test program that prompts the user to enter 10 numbers, invoke this method to return the minimum value, and displays the minimum value. Here is a sample run of the program:

Enter 10 numbers: 1.9 2.5 3.7 2 1.5 6 3 4 5 2

The minimum number is 1.5

```
import java.util.Scanner;
public class HW4number2 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please input 10 numbers to create an array: ");
        double[] myArray = new double[10];
        myArray[0] = input.nextDouble();
        myArray[1] = input.nextDouble();
        myArray[2] = input.nextDouble();
        myArray[3] = input.nextDouble();
        myArray[4] = input.nextDouble();
        myArray[5] = input.nextDouble();
        myArray[6] = input.nextDouble();
        myArray[7] = input.nextDouble();
        myArray[8] = input.nextDouble();
        myArray[9] = input.nextDouble();
        System.out.println("The minimum value of your array is: " +
            min(myArray));
    }
    public static double min(double[] array) {
        double minValue = array[0];
        for (int i = 0; i < array.length; i++) {
            if(array[i] < minValue)
                minValue = array[i];
        }
    return minValue;
    }
}
```



```
Output - second7W (run)
run:
Please input 10 numbers to create an array:
1.9
2.5
3.7
2
1.5
6
3
4
5
2
The minimum value of your array is: 1.5
BUILD SUCCESSFUL (total time: 31 seconds)
```

8.1 (*Sum elements column by column*) Write a method that returns the sum of all the elements in a specified column in a matrix using the following header:

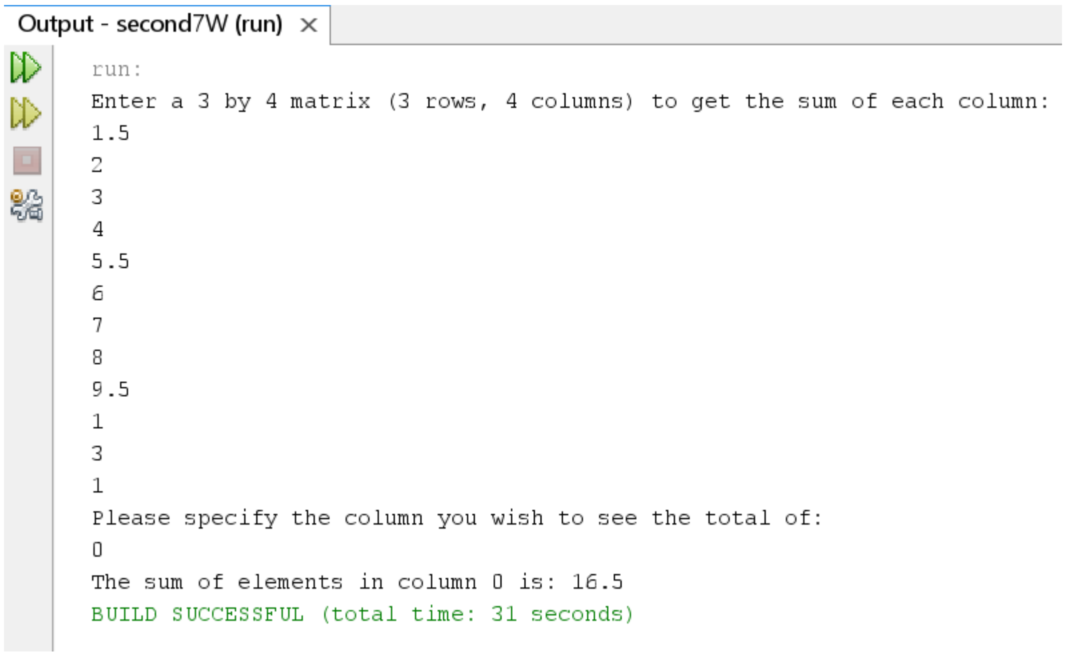
public static double sumColumn (double [][] m, int columnIndex)

Write a test program that reads a 3-by-4 matrix and displays the sum of each column. Here is a sample run:

Enter a 3-by-4 matrix row by row:

```
1.5 2 3 4
5.5 6 7 8
9.5 1 3 1
Sum of the elements at column 0 is 16.5
Sum of the elements at column 1 is 9.0
Sum of the elements at column 2 is 13.0
Sum of the elements at column 3 is 13.0
```

```java
import java.util.Scanner;
public class HW4number3 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Enter a 3 by 4 matrix (3 rows, 4 columns) to get the sum " +
                "of each column: ");
        double[][] m = new double [3][4];
        m[0][0] = input.nextDouble();
        m[0][1] = input.nextDouble();
        m[0][2] = input.nextDouble();
        m[0][3] = input.nextDouble();
        m[1][0] = input.nextDouble();
        m[1][1] = input.nextDouble();
        m[1][2] = input.nextDouble();
        m[1][3] = input.nextDouble();
        m[2][0] = input.nextDouble();
        m[2][1] = input.nextDouble();
        m[2][2] = input.nextDouble();
        m[2][3] = input.nextDouble();

          int columnIndex;

    System.out.println("Please specify the column you wish to see the total of: ");
    columnIndex = input.nextInt();
    System.out.println("The sum of elements in column " + columnIndex + " is: " +
        sumColumn(m,columnIndex));
    }
    public static double sumColumn(double[][]m, int columnIndex) {
        double total = 0;
        if (columnIndex == 0) {
            for (int j = 0; j < m.length; j++){
                total += m[j][0];
            }
        }
        if (columnIndex == 1) {
            for (int j = 0; j < m.length; j++){
                total += m[j][1];
            }
        }
        if (columnIndex == 2) {
            for (int j = 0; j < m.length; j++){
                total += m[j][2];
            }
        }
        if (columnIndex == 3) {
            for (int j = 0; j < m.length; j++){
                total += m[j][3];
            }
        }
        return total;
    }
}
```



```
run:
Enter a 3 by 4 matrix (3 rows, 4 columns) to get the sum of each column:
1.5
2
3
4
5.5
6
7
8
9.5
1
3
1
Please specify the column you wish to see the total of:
0
The sum of elements in column 0 is: 16.5
BUILD SUCCESSFUL (total time: 31 seconds)
```

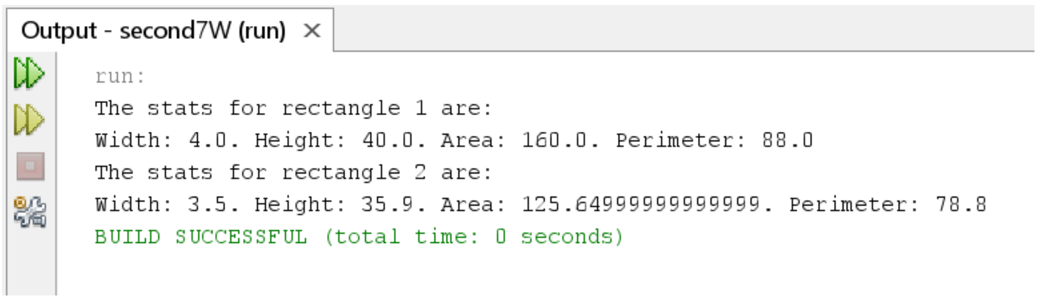
9.1 (*The Rectangle class*) Following the example of the Circle class in Section 9.2, design a class named Rectangle to represent a rectangle. The class contains:

- **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.**

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.

```
public class HW4number4 {
   public static void main(String[] args) {
      Rectangle rec1 = new Rectangle(4,40);
      Rectangle rec2 = new Rectangle(3.5,35.9);
      System.out.println("The stats for rectangle 1 are: ");
      System.out.println("Width: " + rec1.width + ". Height: " + rec1.height +
            ". Area: " + rec1.getArea() + ". Perimeter: " + rec1.getPerimeter());
      System.out.println("The stats for rectangle 2 are: ");
      System.out.println("Width: " + rec2.width + ". Height: " + rec2.height +
             ". Area: " + rec2.getArea() + ". Perimeter: " + rec2.getPerimeter());
   }
}
```

```
class Rectangle {
   double width = 1;
   double height = 1;
   Rectangle(double width, double height) {
      this.width = width;
      this.height = height;
   }
   double getPerimeter() {
      return (2 * width) + (2 * height);
   }
   double getArea() {
      return width * height;
   }
}
```



```
run:
The stats for rectangle 1 are:
Width: 4.0. Height: 40.0. Area: 160.0. Perimeter: 88.0
The stats for rectangle 2 are:
Width: 3.5. Height: 35.9. Area: 125.64999999999999. Perimeter: 78.8
BUILD SUCCESSFUL (total time: 0 seconds)
```

11.1 (*The Triangle class*) Design a class named Triangle that extends GeometricObject defined in Listing 11.1. The class contains:

- **Three double data fields named side1, side2, and side3 with default values 1.0 to denote three sides of a triangle.**
- **A no-arg constructor that creates a default triangle.**
- **A constructor that creates a triangle with the specified side1, side2, and side3.**
- **The accessor methods for all three data fields.**
- **A method named getArea () that returns the area of this triangle.**
- **A method named getPerimeter () that returns the perimeter of this triangle.**
- **A method named toString () that returns a string description for the triangle.**

The formula for computing the area of a triangle is

$$\boldsymbol{s = (side1 + side2 + side3)/2\ ;}$$

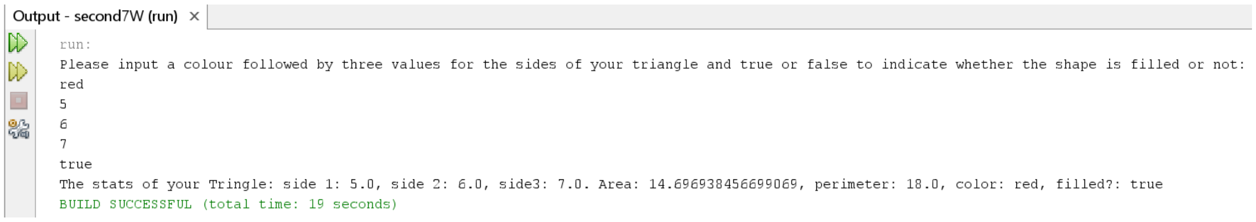
$$\boldsymbol{area = \sqrt{s(s - side1)(s - side2)(s - side3)}}$$

The toString () method is implemented as follows:

return “Triangle: side1 = ” + side1 + “ side2 = ” + side2 + “ side3 = ” + side3;

Write a test program that prompts the user to enter three sides of the triangle, a color, and a Boolean value to indicate whether the triangle is filled. The program should create a Triangle object with these sides and set the color and filled properties using the input. The program should display the area, perimeter, color, and true or false to indicate whether it is filled or not.

```
import java.util.Scanner;
public class HW4number5 {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        System.out.println("Please input a colour followed by three values "
                + "for the sides of your triangle and true or false to indicate " +
                "whether the shape is filled or not: ");

        String color = input.nextLine();
        double side1 = input.nextDouble();
        double side2 = input.nextDouble();
        double side3 = input.nextDouble();
        boolean filled = input.nextBoolean();

        Triangle tri1 = new Triangle(side1, side2, side3, color, filled);

        System.out.println("The stats of your " + tri1.toString() + ". Area: " +
                tri1.getArea() + ", perimeter: " + tri1.getPerimeter() + ", color: " +
                tri1.getColor() + ", filled?: " + tri1.isFilled());



    }
}
    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;
        }

        public String toString() {
            return "created on " + dateCreated + "\ncolor: " + color + " and filled: " +
                    filled;
        }
    }

    class Triangle extends GeometricObject {
        private double side1 = 1.0;
        private double side2 = 1.0;
        private double side3 = 1.0;

        Triangle(){}

        Triangle(double side1, double side2, double side3, String color, boolean filled) {
            this.side1 = side1;
            this.side2 = side2;
            this.side3 = side3;
            setColor(color);
            setFilled(filled);
        }
        public double getSide1() {
            return side1;
        }
        public double getSide2() {
            return side2;
        }
        public double getSide3() {
            return side3;
        }
        public double getArea() {
            double s = (side1 + side2 + side3)/2.0;
            return Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
        }
        public double getPerimeter() {
            return side1 + side2 + side3;
        }
        @Override
        public String toString() {
            return "Tringle: side 1: " + side1 + ", side 2: " + side2 + ", side3: " +
                    side3;
        }
    }
```


Output - second7W (run)

```
run:
Please input a colour followed by three values for the sides of your triangle and true or false to indicate whether the shape is filled or not:
red
5
6
7
true
The stats of your Tringle: side 1: 5.0, side 2: 6.0, side3: 7.0. Area: 14.696938456699069, perimeter: 18.0, color: red, filled?: true
BUILD SUCCESSFUL (total time: 19 seconds)
```