Juan Casanova

Problem 7.10

```
import java.util.Scanner;

public class Q7_10 {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		java.util.Scanner input = new java.util.Scanner(System.in);
		double[] numbers=new double[5];
		int indexofminnumber;
		System.out.println("Enter five numbers:");
		for(int i=0; i<5; i++) {
			numbers[i] = input.nextDouble();
		}
		indexofminnumber = indexOfSmallestElement(numbers);
		System.out.println("The index of smallest number is "+indexofminnumber);

	}
	public static int indexOfSmallestElement(double[]numbers) {

		double minimum = numbers[0];
		int indexOfmin = 0;

		for(int i=1; i<5; i++) {

			if(numbers[i]<minimum) {
				minimum = numbers[i];
				indexOfmin = i;
			}
		}

		return indexOfmin;

	}

}
```

Problem 7.11

```
import java.text.DecimalFormat;
import java.util.Scanner;
```

```
public class Q7_11 {

	public static void main(String[] args) {
		// TODO Auto-generated method stub

		java.util.Scanner input = new java.util.Scanner(System.in);
		DecimalFormat dec = new DecimalFormat("####.###");
		double[] numbers = new double [10];
		double deviationresult;
		double meanresult;
		System.out.println("Enter ten numbers: ");

		for (int i=0; i<10; i++) {
			numbers[i] = input.nextDouble();
		}

		deviationresult = deviation(numbers);
		meanresult = mean(numbers);
		System.out.println("The mean is " + 
dec.format(meanresult));
		System.out.println("The standard deviation is " + 
dec.format(deviationresult));
	}

	public static double mean(double[]numbers) {
		double sum = 0;
		double average;
		for (int i=0; i<10; i++) {
			sum += numbers[i];
		}

		average = sum/10;
		return average;
	}

	public static double deviation(double[]numbers) {

		double sum = 0;
		double sum1 = 0;
		double average = 0;
		double result;

		for (int i=0; i<10; i++) {
```

```
                sum1 += Math.pow((numbers[i]-average),2);
            }
            result = sum1/9;
            result = Math.sqrt(result);
            return result;
        }

}
```

Problem 7.14

```
import java.util.Scanner;

public class q7_14 {

        public static void main(String[] args) {
            // TODO Auto-generated method stub
            java.util.Scanner input = new java.util.Scanner(System.in);
            System.out.println("Enter the 5 Number of elements");

            int[] numbers = new int[5];

            for (int i=0; i<5; i++) {
                numbers[i] = input.nextInt();
            }
            gcd(numbers);

        }
        public static void gcd(int[] numbers) {
            int temp = numbers[0];

            for (int i=1; i<numbers.length; i++) {
                if (temp > numbers[i]) {
                    temp = numbers[i];
                }
            }
            int last = temp;
            for (int re=0; re<numbers.length; re++) {
                boolean result = false;

                for(int j=0; j<numbers.length; j++) {
                    if (numbers[j]%temp !=0) {
                        result = false; break;
                    } else {
```

```
                                        result = true;
                                }
                        }
                        if (result) {
                                break;
                        } else {
                                temp--;
                        }
                }
                System.out.println("The GCD is " + temp);
        }

}
```

Problem 7.19

```
import java.util.Scanner;


public class Q7_19 {

        public static void main(String[] args) {
           int size = 100;

              Scanner input = new Scanner(System.in);
              System.out.print("Enter list: ");
              size = input.nextInt();
              int[] numbers = new int[size];
              for (int i = 0; i < size; i++) {
                 numbers[i] = input.nextInt();
              }
              if (isSorted(numbers)) System.out.print("The list is already sorted.\n");
              else System.out.print("The list is not sorted.\n");

           }

           public static boolean isSorted(int[] numbers) {

              for (int i = 0; i < numbers.length - 1; i++) {

                 if (numbers[i] > numbers[i + 1]) return false;

              }
              return true;
```

```
            }
            public static void printArray(int[] array, int numberPerLine) {

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

                           System.out.printf("%4d ", array[i]);
                           if ((i + 1) % numberPerLine == 0) System.out.println("");
                        }
            }
        }
```

Problem 8.1

```
import java.util.Scanner;

public class q8_1 {

         public static void main(String[] args) {
                  System.out.print("Enter a 3 X 4 matrix: ");
      Scanner input = new Scanner(System.in);

      // read user input: 3 by 4 matrix
      double[][] matrix = new double[3][4];
      for (int i = 0; i < matrix.length; i++)
         for (int j = 0; j < matrix[i].length; j++)
            matrix[i][j] = input.nextDouble();

      for (int i = 0; i < matrix[0].length; i++) {
         System.out.println("Sum of the elements at column" + i +" is " + sumColumn(matrix, i));
      }

   }

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

      double total = 0;

      for (int i = 0; i < m.length; i++) {
         total += m[i][columnIndex];
      }
      return total;
   }

   public static void displayMatrix(double[][] matrix) {
        for (int row = 0; row < matrix.length; row++) {

            for (int column = 0; column < matrix[row].length; column++) {
                System.out.printf("%5.0f ", matrix[row][column]);
            }
            System.out.printf("\n");
        }
    }

}
```

Problem 8.5

```
import java.util.Scanner;
public class Q8_5 {

            public static void main(String[] args) {
                        Scanner input = new Scanner(System.in);
        System.out.print("Enter 3x3 matrix 1: ");
        double[][] matrix1 = new double[3][3];
        for (int i = 0; i < matrix1.length; i++)
            for (int k = 0; k < matrix1[i].length; k++)
                matrix1[i][k] = input.nextDouble();


        System.out.print("Enter 3x3 matrix 2: ");
        double[][] matrix2 = new double[3][3];
        for (int i = 0; i < matrix2.length; i++)
            for (int k = 0; k < matrix2[i].length; k++)
                matrix2[i][k] = input.nextDouble();

        double[][] addedMatrix = addMatrix(matrix1, matrix2);

        for (int i = 0; i < matrix1.length; i++) {

            for (int k = 0; k < matrix1[i].length; k++) {
                System.out.printf("%2.1f ", matrix1[i][k]);
                if (i == 1 && k == 2) System.out.printf("%2s ", " + ");
                else System.out.printf("%3s ", " ");
            }
            for (int k = 0; k < matrix2[i].length; k++) {
                System.out.printf("%2.1f ", matrix2[i][k]);
                if (i == 1 && k == 2) System.out.printf("%2s ", " = ");
                else System.out.printf("%3s ", " ");
            }
            for (int k = 0; k < addedMatrix[i].length; k++) {
                System.out.printf("%4.1f ", addedMatrix[i][k]);
            }
            System.out.println("");

        }
    }

    public static double[][] addMatrix(double[][] a, double[][] b) {

        double[][] addedMatrix = new double[a.length][a[0].length];

        for (int i = 0; i < a.length; i++) {
            for (int k = 0; k < a[0].length; k++) {
                addedMatrix[i][k] = a[i][k] + b[i][k];
            }
        }

        return addedMatrix;

            }

}
```

Problem 8.7

```
public class Q8_7 {

            public static void main(String[] args) {
                        double[][] points = new double[][] {
             {-1, 0, 3},    {-1, -1, -1},
             {4, 1, 1},     {2, 0.5, 9},
             {3.5, 2, -1},  {3, 1.5, 3},
             {-1.5, 4, 2},  {5.5, 4, -0.5}
    };

    int p1 = 0, p2 = 1; // Initial two points
    double shortestDistance = distance(points[p1][0], points[p1][1], points[p1][2],
          points[p2][0], points[p2][1], points[p2][2]); // Initialize shortestDistance
    // Compute distance for every two points
    for (int i = 0; i < points.length; i++) {
      for (int j = i + 1; j < points.length; j++) {
        double distance = distance(points[i][0], points[i][1], points[i][2],
              points[j][0], points[j][1], points[j][2]); // Find distance

        if (shortestDistance > distance) {
          p1 = i; // Update p1
          p2 = j; // Update p2
          shortestDistance = distance; // Update shortestDistance
        }
      }
    }

    // Display result
    System.out.println("The closest two points are " +
        "(" + points[p1][0] + ", " + points[p1][1] + ") and (" +
        points[p2][0] + ", " + points[p2][1] + ")");
  }

  public static double distance(
      double x1, double y1, double z1, double x2, double y2, double z2) {
    return Math.sqrt( Math.pow((x2 - x1), 2) + Math.pow((y2 - y1), 2) + Math.pow(z2 - z1, 2));
  }
}
```

Problem 8.10

```
public class Q8_10 {

        public static void main(String[] args) {
                // TODO Auto-generated method stub
                int[][] matrix = new int[4][4];

      int largestRI = 0;
      int largest = -1;
      for (int i = 0; i < matrix.length; i++) {
        int rowCount = 0;
        for (int k = 0; k < matrix[i].length; k++) {
          matrix[i][k] = (int)(Math.random() * 2);
          rowCount += matrix[i][k];
        }
        if (rowCount > largest) {
          largestRI = i;
```

```
                largest = rowCount;
            }
        }

        int largestCI = 0;
        largest = -1;
        for (int k = 0; k < matrix[0].length; k++) {
            int columnCount = 0;
            for (int i = 0; i < matrix.length; i++) {
                columnCount += matrix[i][k];
            }
            if (columnCount > largest) {
                largest = columnCount;
                largestCI = k;
            }

        }

        for (int i = 0; i < matrix.length; i++) {
            for (int k = 0; k < matrix[i].length; k++) {
                System.out.printf("%d", matrix[i][k]);
            }
            System.out.printf("\n");
        }
        System.out.println("The largest row index: " + largestRI);
        System.out.println("The larges column index: " + largestCI);
    }


}
```