UNK CSIT 150
Lab 1: Two-Dimensional Arrays

Objectives

- Practice implementing two-dimensional arrays
- Practice implementing files & other CSIS 130 concepts

General Requirements

In this lab, your code must adhere to the following programming style:

- Use *indention* to show the logical structure of your code
- Use blank lines to separate code blocks and use methods to make your code modular.
- Give each code block a comment and use JavaDoc documentation for each method.
- Give your variables intuitive names and start with a lower case letter.
- Give constants intuitive names that are in all capital letters.
- Opening and closing braces should be aligned vertically to each other.

Programming Practice

Your program will read temperature data from a file (structured as shown below) into a two-dimensional array (of doubles). You will also want a one-dimensional array of strings for the month names. The program should have the following methods. Additional methods should be used as needed to solve this problem.

- **getRowAverage**. This method should accept a single-dimensional array as its first argument. The method should return the average of the values in the array. Use this method to calculate the monthly average for each month.

- **getColumnAverage**. This method should accept a two-dimensional array as its first argument and an integer as its second argument. The second argument should be the subscript of a column in the array. The method should return the average of the values in the specified column. Use this method to calculate the average for each year.

- **getMonthValues**. This method should accept a two-dimensional array as its first argument and an integer as its second argument. The second argument should be the subscript of a month (row) in the array. The method should return the values in the specified row as a String. Use this method to output the values for each month.

- **readFile**. This method returns a two-dimensional array, and has a File as its argument. It will return the data contained in the file newTempData.csv as an array.

The **Temperature data** is stored in a file named newTempData.csv and sample data is shown below. Each line contains the name of the month and 6 values for the temperature on the first of the month for 6 years. The sample output is shown on the next page.

```
January,17.54,-9.72,27.44,1.13,5.99,-14.14
February,20.8,-14.67,20.53,-0.97,-4.21,-17.07
March,38.35,19.44,36.34,3.45,20.39,12.42
April,48.3,26.29,59.56,53.53,58.09,19.37
May,48.45,78.59,77.64,56.37,75.27,51.75
June,53.21,76.25,63.54,80.5,80.06,80.71
July,94.06,83.61,77.78,75.73,94.01,64.22
August,93.31,92.29,80.34,100.82,100.27,94.89
September,77.32,58.59,77.51,69.51,60.96,55.9
October,50.18,62.81,46.5,46.58,45.67,67.15
November,47.09,32.45,47.21,26.35,46.91,20.07
December,27.48,28.42,5.02,8.67,31,25.71
```

Hint: Using a scanner, read each line in the file. Then split the line based on the comma data.

```
Scanner inputScanner = new Scanner(inputFile);
…
while (inputScanner.hasNext){
        String monthData = inputScanner.nextLine();
        String[] splitData = monthData.split(“,”);
// now put the split data into the correct array spots.
…
```

Report to the Instructor

Before you leave, show the instructor the source code and demonstrate how the code runs.

Sample output for the data shown above:

```
Month            Temperatures                                        Average
January:         17.54   -9.72   27.44   1.13    5.99    -14.14     4.71
February:        20.80   -14.67  20.53   -0.97   -4.21   -17.07     0.74
March:           38.35   19.44   36.34   3.45    20.39   12.42      21.73
April:           48.30   26.29   59.56   53.53   58.09   19.37      44.19
May:             48.45   78.59   77.64   56.37   75.27   51.75      64.68
June:            53.21   76.25   63.54   80.50   80.06   80.71      72.38
July:            94.06   83.61   77.78   75.73   94.01   64.22      81.57
August:          93.31   92.29   80.34   100.82  100.27  94.89      93.65
September:       77.32   58.59   77.51   69.51   60.96   55.90      66.63
October:         50.18   62.81   46.50   46.58   45.67   67.15      53.15
November:        47.09   32.45   47.21   26.35   46.91   20.07      36.68
December:        27.48   28.42   5.02    8.67    31.00   25.71      21.05
Average column (Yearly) temperatures
                 51.34   44.53   51.62   43.47   51.20   38.41
Thank you.
```

Lab 1: Multidimensional Arrays

Name(s): _____

Evaluation

Requirement	**Possible Points**	**Points Received**	**Comments**
getRowAverage Method	2		
getColumnAverage Method	2		
getMonthValues Method	2		
readFile Method	2		
Output formatted properly	2		

Programming style	**Possible Points**	**Points deducted**	
Inconsistent indentation	**-1**		Indent at least 3 spaces inside each brace
Poor use of white space	**-1**		Too many or too few lines between statements
Heading documentation, includes programmer names, date, algorithm, basic purpose	**-2**		
All sources not cited (Remember to cite all code used, even class demo code.)	**-1**		
JavaDocs not used on each method	**-1**		
Poor variable names	**-1**		Should not start with uppercase. No single letter variables.
Poor structure/logic issues	**-2**		
Program specifications			
Too many arrays defined in main method for data	**-1**		Only two arrays should be created in the main method, one for the data and one for the name of the months.

Instructor must sign-off before end of lab period. If you do not get it done by this deadline, you have until one week to submit your work with a 0.4 point late penalty.

Instructor signature: _____