COSC 1436: Programming Fundamentals I
(Assignment 4)

- **DUE Time and Date**: **Midnight, Friday, 04/14/2017**
- **What to submit:** **Array.java**
LastNameFirstNameArrayTestDriver.java
LastNameFirstNameTexansStatistics.java

Note 1: Unless otherwise mentioned, you are asked to upload ONLY your java source files through blackboard. Email submission is not accepted, because of confusion in grading.

Note 2: If your programs contain any syntactical errors, no points will be given. Thus, please make sure your programs are properly compiled with computers at the CS labs, not only in your laptop or desktop environments.

Note 3: No late submission will be accepted, thus keep the deadline.

Note 4: Grading will be divided into two categories, formatting and logic, where formatting compromise 25% of your total grade. Formatting will be based on the following rules.

Rule 1: Naming is an important issue in Java. Not only you should you define meaningful variable names, but should also give appropriate names for the physical java file, which must be the same as the public class name that you edit.

Unless otherwise mentioned, you will follow **the industry standard for Java naming convention**:

(1) Java Classes start in uppercase and each individual word in the class name is capitalized;

(2) All Java methods and variables start in lowercase and each individual word in the method and variable is capitalized;

(3) Each final variable (known as a constant) should be written in all uppercase.

Rule 2: There should be a space around all operators (e.g., 3 + 5, not 3+5). In addition, spacing with regards to parentheses should be consistent.

Rule 3: In addition to the Java naming conventions, you are asked to add your name in front of each class name like **LastNameFirstNameClassName.java.**

For instance, if your name is “John Doe” and the class name is “RightTriangle”, then the class name in your source code should be “DoeJohnRightTriangle” and your corresponding physical file name should be “DoeJohnRightTriangle.java”.

Rule 4: Everything nested inside of an open brace should be indented with regular-sized spaces (say, 4 or 8 spaces). The open brace for functions and classes should (1) come at the end of the line and be preceded by a space like

```
public  class  DoeJohnRightTriangle  {
      public  static  void  main( String args[] ) {
      }
}
```

or (2) start with the new line as shown below:

```
public  class  DoeJohnRightTriangle
{
      public  static  void  main( String args[] )
      {
      }
}
```

Rule 5: Always type block Javadoc comments to include title of the project, program’s purpose, your name, the date, and the version number as in the lectures or in the labs. For example,

```
/*****************************************************************
@Title:     LastNameFirstNameClassName
@Purpose:   To verify the edit, compile, execute function in Textpad
@Author:    (your last & first name)
@Date:      (today’s date)
@Version:   1.0
*****************************************************************/
```

Question 1 (50 points): Array & Array Tester

We want to implement an **`Array`** class that contains the following field and methods.

`Array`
`- array:double[]`

`+ setArray(arr:double[]):void`
`+ isInIncreasingOrder():boolean`
`+ inInDecreasingOrder():boolean`
`+ getHightest():int`
`+ getLowest():int`
`+ getTotal():double`
`+ getAverage():double`
`+ getAarray():double[]`

Note that the **`Array`** class doesn't have any constructors.

`+ setArray(arr:double[]):void`

It copies the argument **`arr`** into the class's field **`array`**.

`+ isInIncreasingOrder():boolean`

It returns **`true`** if the **`array`** is sorted in increasing order.

`+ inInDecreasingOrder():boolean`

It returns **`true`** if the **`array`** is sorted in decreasing order.

`+ getHightest():int`

It returns the highest value from the **`array`**.

`+ getLowest():int`

It returns the lowest value from the **`array`**.

`+ getTotal():double`

It returns the total of the values in the **`array`**.

`+ getAverage():double`

It returns the average of the values in the **`array`**.

You need to test the functionality of the **`Array`** class that you have created. Write a driver class, named **`ArrayTestDriver`**, to test all of the methods of the **`Array`** class as follows:

(1) Ask a user to enter the size of an array, and create an array with the size entered.
(2) Run a **`for`** loop to ask the user to enter elements of the array, and assign the values to the array.
(3) Pass the created array to the **`Array`** class's **`setArray`** method.
(4) Check if the array is sorted in increasing or decreasing order using the **`Array`** class's method(s), and display the result.
(5) Display the highest value and lowest value of the array using the **`Array`** class's method(s).
(6) Display the total value of the array using the **`Array`** class's method(s).
(7) Display the average value and lowest value of the array using the **`Array`** class's method(s).

Your file will have the following headers:

```
/**********************************************************************
@Title:     Array.java
@Purpose:   To get familiar with class, array, and their usage
@Author:    (your last first name)
@Date:      (today's date)
@Version:   1.0
**********************************************************************/
/**********************************************************************
@Title:     LastNameFirstNameArrayTestDriver.java
@Purpose:   To get familiar with class, array, and their usage
@Author:    (your last first name)
@Date:      (today's date)
@Version:   1.0
**********************************************************************/
```

Question 2 (50 points): Texan's Statistics

Write a program that prompts the user to enter the Houston Texans' first eight regular season scores. The scores should be saved in two different arrays, named **`texansScore[]`** and **`opponentsScore[]`**. The program should display:

a) the percentage of games won by the Houston Texans,
b) the highest score of Houston Texans and the corresponding game number,
c) the highest total score (Texans' and Opponents' combined) of all the games and the corresponding game number as well.

For example:

If the user enters the following data for Texans, and

37	39	47	35	16	34	24	19

if the user enters the following data for Opponent's Score

20	56	18	28	20	26	18	10

Output:

```
Winning Percentage is: 75%
Highest Houston Texans score was 47 in the 3rd game.
The total score of the highest scored game was 95 in the 2nd game.
```

Your file will have the following documentation header:

```
/**********************************************************************
@Title:     LastNameFirstNameTexansStatistics
@Purpose:   To get familiar with single dimensional arrays
@Author:    (your last first name)
@Date:      (today's date)
@Version:   1.0
**********************************************************************/
```