TCES 203
Programming Practicum
Extra Credit Java Assignment
15 Points
This homework must be done individually

This assignment contains several problems that test your understanding of Java concepts. You may submit some of the problems instead of all the problems. The problems that you solved must be complete to get any credit.

1. (3 Points) Write a static method named printGrid that accepts two integer parameters rows and cols. The output is a comma-separated grid of numbers where the first parameter (rows) represents the number of rows of the grid and the second parameter (cols) represents the number of columns. The numbers count up from 1 to (rows x cols). The output are displayed in column-major order, meaning that the numbers shown increase sequentially down each column and wrap to the top of the next column to the right once the bottom of the current column is reached. Assume that rows and cols are greater than 0. Write you code in **Grid.java** that contains printGrid method and the main method.

   Here are some example calls to your method and their expected results:

Call:	`printGrid(3, 6);`	`printGrid(5, 3);`	`printGrid(4, 1);`	`printGrid(1, 3);`

Output:	`1, 4, 7, 10, 13, 16`
	`2, 5, 8, 11, 14, 17`
	`3, 6, 9, 12, 15, 18`
		`1, 6, 11`
		`2, 7, 12`
		`3, 8, 13`
		`4, 9, 14`
		`5, 10, 15`
			`1`
			`2`
			`3`
			`4`
				`1, 2, 3`

2. (3 Points) Write a static method named minGap that accepts an integer array as a parameter and returns the minimum 'gap' between adjacent values in the array. The gap between two adjacent values in a array is defined as the second value minus the first value. For example, suppose a variable called array is an array of integers that stores the following sequence of values.

   int[] array = {1, 3, 6, 7, 12};
   The first gap is 2 (3 - 1), the second gap is 3 (6 - 3), the third gap is 1 (7 - 6) and the fourth gap is 5 (12 - 7). Thus, the call of minGap(array) should return 1 because that is the smallest gap in the array. Notice that the minimum gap could be a negative number. For example, if array stores the following sequence of values:

   {3, 5, 11, 4, 8}

The gaps would be computed as 2 (5 - 3), 6 (11 - 5), -7 (4 - 11), and 4 (8 - 4). Of these values, -7 is the smallest, so it would be returned.

This gap information can be helpful for determining other properties of the array. For example, if the minimum gap is greater than or equal to 0, then you know the array is in sorted (nondecreasing) order. If the gap is greater than 0, then you know the array is both sorted and unique (strictly increasing).

If you are passed an array with fewer than 2 elements, you should return 0.
Write you code in **Gap.java** that contains minGap method and the main method.

3. (5 Points) Write a class called Line that represents a line segment between two Points. You must use the Point class provided by Java in the java.awt package. Your Line objects should have the following methods:

   public Line(Point p1, Point p2)
   Constructs a new line that contains the given two points.

   public Point getP1()
   Returns this line's first endpoint.

   public Point getP2()
   Returns this line's second endpoint.

   public String toString()
   Returns a string representation of this line, such as "[(22, 3), (4, 7)]".

   Submit **Line.java** and **LineTest.java** (that contains the main method).

4. (4 Points) Write a static method minToFront that takes an ArrayList of integers as a parameter and that moves the minimum value in the list to the front, otherwise preserving the order of the elements. For example, if a variable called list stores the following values: {3, 8, 92, 4, 2, 17, 9} and you make this call: minToFront(list); it should store the following values after the call: {2, 3, 8, 92, 4, 17, 9} You may assume that the list stores at least one value. Write you code in **Front.java** that contains the methods, minToFront and the main method.

Submission and Grading:

Submit all java files compressed into a zip file with the name ExtraCredit.

There will be points taken off for not following the conventions listed in this document regarding submissions, outputs and naming conventions.

You are required to properly indent your code and will lose points if you make significant indentation mistakes. See the textbook for an explanation and examples of proper indentation.

Give meaningful names to functions and variables in your code. Localize variables whenever possible -- that is, declare them in the smallest scope in which they are needed.

Include a comment at the beginning of your program with basic information and a description of the program **and include a comment at the start of each method**. Your comments should be written in your own words and not taken directly from this document. Write comments within methods to explain the flow or any obscure code. Provide comments for the methods as well as for the class definition. Make sure that every file has a header comment.

You should include a comment at the beginning of your program with some basic information and a description of the program, as in:

```
// Menaka Abraham
// 3/30/15
// 203
// Assignment #1
//
// This program will...
```