CS1073 - Assignment #11 – Fall 2020

Submission Deadline: Friday, December 4th before 12:00 NOON (Atlantic Daylight Time Zone) in the Assignment 11 dropbox in Desire2Learn. (Read the submission instructions at the end of this document carefully).

The purpose of this assignment is to help develop your understanding of loops, and to have you begin working with one-dimensional arrays of primitive values.

Side Note: When repetition is required, we can employ loops, or we can use a technique called *recursion* (which you will learn about in CS1083). Your answers to these assignment questions should use loops, not recursion.

This assignment is to be done individually. If you have questions, direct them to a tutor/assistant during a help session in the "Faculty of Computer Science Student Success Centre" team or to your course instructor.

As always, include a Javadoc comment for each of your classes (with @author information).

I. Preliminary Nested-Loop Exercises:

a) Tracing
(No marks for part a); however, this will help you in the remaining questions.)

What is the output of the following code?

```
public class Pattern {
   public static void main (String[] args) {
      for (int i=10; i>0; i--) {
         for (int j=1; j<=i; j++) {
            System.out.print('*');
         }
         System.out.println();
      }
   }
}
```

b) Using Nested Loops:

Re-write the above code so that it produces the following output:

```
         *
        **
       ***
      ****
     *****
    ******
   *******
  ********
 *********
**********
```

PLEASE NOTE the following restriction: In your solution, you may only print single characters (or newlines); you may NOT pass a String to the print or println methods.

Include a Javadoc comment for the revised class (with @author information), and capture sample output for your solution.

II. Gathering and Summarizing Data:

Write a Java application to read a list of exam scores from the user, one by one. Each exam score should be a whole number in the range 0-100. A negative exam score is used as a sentinel value (i.e. this marks the end of input). Any exam scores above 100 are invalid and an appropriate error message should be displayed; the user would then be prompted to enter another (valid) exam score and the program would continue.

The program should output class statistics. Specifically, it should display the number of A grades, the number of B grades, the number of C grades, the number of D grades and the number of F grades. Statistics should be based on the following ranges:

```
85 - 100  =  A
70 -  84  =  B
55 -  69  =  C
45 -  54  =  D
 0 -  44  =  F
```

Test your program with various different grades, including invalid input (i.e. integer values above 100), and capture sample output.

III. Horizontal Histogram:

A "histogram" is a simple graph that can be used to display two-dimensional data. For example, the following histogram displays the average rainfall (in cm) for each month of the year for a particular city (Note: the months are listed along the y-axis, and the rainfall amounts on the x-axis):

```
Month
 1    |***
 2    |*****
 3    |**********
 4    |***********************
 5    |********************************
 6    |************
 7    |***********
 8    |****************
 9    |*************
10    |*********
11    |*******
12    |*****
      ===================================================
      +         +         +         +         +         +
      0         10        20        30        40        50
                              cm
```

Make a copy of your solution to question II and then change the code so that the class statistics are plotted on a histogram. List the possible grades along the vertical (y) axis, and draw the histogram going outwards across the screen, with the totals on the x-axis. Your histogram should use the format shown in the sample histogram above (with appropriate values and labels along each axis.) Note: For this question, you may assume that there will never be more than 30 students in each grade category.

Test your program with a list of at least 25 grades and capture sample output.

IV. Vertical Histogram (a bit more challenging):

Make another copy of your solution to question II and revise the program once again to display the results using a vertical histogram (with appropriate labels). This time, list the possible grades along the horizontal (x) axis, and list the totals along the vertical (y) axis (with 0 at the bottom and the numbers increasing as they go up that axis).

Again, test your program with at least 25 grades and capture sample output.

V. Working with One-Dimensional Arrays of Primitive Values:

a) Write a class named **`IntArrayUtil`**. This class will only contain the following three *static* methods:

`public static int[] append (int[] arrA, int[] arrB)`

This method appends one integer array after another and returns that as a new array. (The parameters themselves are not altered.)

For example, if `testArray1` contains:

1	4	9	16

and `testArray2` contains:

9	7	4	9	11

Then `IntArrayUtil.append(testArray1, testArray2)` returns a new array containing:

1	4	9	16	9	7	4	9	11

`public static int[] reverse (int[] arr)`

This method reverses the sequence of elements in an integer array and returns that in a new array. (The parameter itself is not altered.)

For example, if `reverse` is called with an array containing:

1	4	9	16	9	7	4	9	11

Then it returns a new array containing:

11	9	4	7	9	16	9	4	1

`public static int alternatingSum (int[] arr)`

This method computes and returns the alternating sum of all elements in the integer array that is passed in via its parameter.

For <u>example</u>, if `alternatingSum` is called with an array containing:

1	4	9	16	9	7	4	9	11

then it computes:

1 - 4 + 9 - 16 + 9 - 7 + 4 - 9 + 11 = -2 (and returns -2)

Recall: A static method is one that is called directly, using the class name. No object needs to be constructed to call a static method. (The methods in the Math class are all static.)

Notes:

- You should not call any methods from the Java API in your solution to this question. (Solve the problems yourself, using loops.)
- For all of the methods in the IntArrayUtil class, you may assume that the input arrays are all full (i.e. no partially-filled arrays).
- Write a javadoc comment for this class and for each of the 3 methods. Include @author, @param & @return tags where appropriate.

b) Write a test driver that thoroughly tests the three methods from part a. Use a few different test cases (arrays with different lengths & different contents). Note: Your test driver should be in a separate class and file.

In your test driver, print out the *contents* of the input arrays (the ones that you pass in as parameters), and also print the *contents* of the arrays that are returned from the first two methods as well as the number that is returned from the third method.

Be sure to include a Javadoc comment for your class (with @author information).

Capture sample output to hand in for marking.

Submission instructions are on the next page…

Your electronic assignment submission (submitted via Desire2Learn) will consist of two files:

i. a written report. This should begin with a title page; just as we described in Assignment #1, your title page should include: the course (CS 1073), your section (FR01A, FR02A, FR03A or FR04A), the assignment number, your full name, and your UNB student number. That should be followed by five sections, with each part clearly identified with a section heading. Include the source code and sample output for each question. (Only Question I part a) does not need to be handed in for marking.)

   (Aside: As always, your source code should contain javadoc comments, but you do not need to include the .html files.)

   This written report should be prepared using a word processor; we recommend using Microsoft Word (i.e. create a .docx file for your report). Copy & paste your java source code & required output into the report document. Add appropriate headings for each part. Fix up the formatting where necessary, adjusting line breaks & page breaks to ensure that your document is easy to read. Use a monospaced font for your code to maintain proper indentation.) Once the report is complete and you've checked it all over, save the .docx file for your own records, and then **save a second copy in pdf format for submission**. (Note: Be sure to open that file in a pdf viewer to verify that the pdf was generated correctly.) The **SINGLE pdf file** containing your report will be submitted to the appropriate assignment drop box on Desire2Learn. (It is important that you submit a pdf file and NOT the original Word document. This pdf will allow the marker to write comments directly on your work to give you better feedback.)

   Note: Please name this report as follows: **YourName_As11_Report.pdf**

ii. an archive file **(.zip)** that contains your Java source code and output for this assignment. Make sure that your archive includes all .java files (in case the marker wishes to compile & run your code to test it). You should not include the report document or the .class files in your archive. This archive should be submitted as a **single file** to the appropriate drop box on Desire2Learn.

   Note: Please name this archive file as follows:
   YourName_As11_Archive.zip

End of Assignment 11

Maintained by Natalie Webber