**CS 1450 Data Structures and Algorithms – Spring 2020**

**Assignment #1**

Due Date: **Jan 29, 2020** at 1:40pm (MW class), **Jan 30, 2020** at 1:40 (TR class)

Purpose: Get your mind back into writing code. Review of loops, arrays, and files.

Effort: Individual

Points: **50**

Deliverables: Upload a **.zip** file with **ONLY** your source code (.java file) to Canvas by due date.

**Hand in a hardcopy version of the code and your design notes at beginning of class.**

**Assignment Description**

Arrays were the first data structure introduced in your beginning Java class and knowing how to manipulate arrays is an important skill. We’ll review arrays by filling an array with random values and then performing a few operations on the array. You may also have learned to use files which is another important skill we will use throughout the semester. To review files (or learn for the first time), we will perform a simple writing to and reading from a file.

**Specifications**

1. If using Eclipse, create a Java **project** called **CS1450**
   1. File -> New -> Java Project
2. Create a Java **class** (File->New->Class) within that project called **LastNameFirstNameAssignment1** 
   1. Please use this naming convention on all assignments.
   2. For example, I would name my file GonzalezMickeyAssignment1
3. The following modules on Canvas contain several helpful documents:
   1. ***Review: Object Oriented Programming and File*** for help with files.
   2. ***Programming Assignments Policy*** for help with assignment requirements.
   3. ***Design Notebook*** for help with design notebook and an example for this assignment.
4. For this assignment, all code can be in **main() –** that is **–** methods and classes are not required.

**Task 1:** perform the following array processing tasks:

1. Generate and display two random numbers – ***size1*** and ***size2*** - between 1 and 10.
   1. The values ***size1*** and ***size2*** will be used in step 2 as the size of each array.
2. Generate ***size1*** random integers between 1 and 25 and write them to the 1st array
3. Generate ***size2*** random integers between 1 and 25 and write them to the 2nd array
4. Sort each array using the **sort** method in the **Arrays** class
5. After each array is sorted, display the values in each array.

**Task 2:**  perform the following file and array-based tasks:

1. In your code create a file named **assignment1.txt**.
   1. **Correct:**  File fileName = **new** File("assignment1.txt");
      1. **Do not** include any path names when creating the file.
      2. The file will be placed in your current working directory.
   2. **Incorrect:**  File fileName = **new** File("C:/Dev/assignment1.txt");
2. Open the file for writing.
   1. PrintWriter resultsFile = **new** PrintWriter (fileName);
3. Write the values in the two arrays in **sorted order** to the file.
   1. See the output below where it shows “Writing values to file” for an example.
   2. Display each value as you write it to the file
4. Close the file.

**Task 3:**  perform the following file and array-based tasks:

1. Reopen the file for reading.
   1. Scanner readFile = **new** Scanner (fileName);
2. Read values from file and place into an array, **removing duplicates** as you read the values.
3. Display the array containing the sorted list of values without duplicates.

**Must Do and Tips**

**Must Do**

* You must use an array, not an array list.

**Must Not Do**

* Do not include path names for the file. Use only the name **assignment1.txt**
  + File fileName = **new** File("assignment1.txt");
* To create the file of sorted values, **DO NOT** move the values in the two arrays into another array, sort, then write to the file. That process requires little array manipulation.
  + Instead, you must write code that walks through the arrays and writes one value at a time to the file – that is – manually combine the array values into the file.
  + This process requires you to perform array manipulation.
  + See the tips below for one way to write this code.

**Tips**

* The **assignment1.txt** file will be overwritten each time your code is run.
* Merging the two arrays **manually** into the file requires two steps:
  + Step 1: write a while loop to take values from the arrays in sorted order and write them to the file.
    - This while loop completes when all values in **one** of the arrays have been written to the file.
    - Since the array sizes are randomly created, in most cases the arrays will be different sizes. This means, in most cases, one array will have all it values written to the file while the other array still contains values that need to be written to the file.
  + Step 2: after the above while loop completes, determine which array still contains values and write those remaining values to the file.

**while** (more values in Array1 to write to file) {

// Write remaining values in array1 to file

}

**while** (more values in Array2 to write to file) {

// Write remaining values in array2 to file

}

**Output**

Your output should look like the following except with different random numbers.

**Output – Example**

size1 = 4

size2 = 3

First array with 4 sorted random values

---------------------------------------

7

8

10

24

Second array with 3 sorted random values

---------------------------------------

12

12

25

Writing values to the file

--------------------------

Writing: 7

Writing: 8 Note: when writing the values to the

Writing: 10 file, they are written in sorted order.

Writing: 12

Writing: 12 When the values are read from the file and placed

Writing: 24 into a new no-duplicates array, 12 shows up once.

Writing: 25

Array with no duplicate values

------------------------------

7

8

10

12

24

25