CS1083 Assignment # 2 – Fall 2021

Due: Wednesday, September 29th before 11:00pm (Atlantic), submitted in the Assignment 2 submission folder in Desire2Learn. (Read the submission instructions at the end of this document carefully).

The purpose of this assignment is to provide you with more practice with loops and arrays.

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.

---

1. Operations on Sorted Arrays:

The purpose of this exercise is to provide more practice working with loops and arrays. Here you will be writing three methods that operate on sorted arrays: finding the number of duplicate elements when comparing two arrays, merging two sorted arrays into one sorted array, and adding an element to an existing array such that the ordering is preserved. The arrays you will be creating and working with will be filled arrays (not partially filled arrays).

You are provided with three java source files in D2L. One, **Item.java**, is a class that represents an item in a customer's wish list and provides methods to access attributes, change the priority of an item, and calculate the difference between two items' SKUs. The second, **WishList.java**, represents a customer's complete wish list, and provides methods to create a wish list given an array of items or create one by reading items using a Scanner object. In the latter case the input format consists of a line with the number of items, followed by a line for each item containing values separated by commas (see example below). The inventory list is sorted by ascending value of the stock keeping numbers (SKU) of the items. Finally, **TestWishList.java** is a simple test driver. Note: there will not be duplicate items within a single customer's wish list.

Your task is to implement three methods in **WishList.java**, and test them appropriately.

1. **public int findUnique(WishList other)**

   Returns the number of items in one wish list but not in the other. Note: The fact that the items in each wish list are sorted makes finding duplicates easier than if items were stored in arbitrary order.

2. **public boolean addItem(Item itemIn)**

   Adds an item to this wish list, in the appropriate location so that the wish list remains sorted in ascending order. **Duplicates are not allowed in individual lists**. Returns true if the item was added successfully, false if not.

3. **public WishList merge(WishList other)**
   Merges this wish list with another one, producing a new sorted wish list. **If the same item appears in both wish lists, then it appears twice in the merged wish list** (i.e., duplicates are not removed). Note: use the fact that the lists are each already in order to perform the merge efficiently.

For full marks, we are looking for an efficient solution for each of these methods (that takes the fact that each list is sorted into account – Use this to your advantage!).

Example

Consider the following input:

```
4
11039926010,Digital Kitchen Scale,3
11798411010,KitchenAid Stand Mixer,1
24179114710,Autumn Plaid Tablecloth,2
96796133410,Tan Cotton Blanket,2
4
11781701910,Cast Iron Round Griddle,2
11798009510,Espresso Machine,1
11798112010,NutriBullet Blender,1
11798411010,KitchenAid Stand Mixer,2
```

When the test driver is run with this input, the following is produced:

```
List 1:
11039926010     Digital Kitchen Scale   3
11798411010     KitchenAid Stand Mixer  1
24179114710     Autumn Plaid Tablecloth 2
96796133410     Tan Cotton Blanket      2
```

```
Item added to List 1:
11039926010     Digital Kitchen Scale   3
11798411010     KitchenAid Stand Mixer  1
11881701910     Rice Cooker & Steamer   2
24179114710     Autumn Plaid Tablecloth 2
96796133410     Tan Cotton Blanket      2

List 2:
11781701910     Cast Iron Round Griddle 2
11798009510     Espresso Machine        1
11798112010     NutriBullet Blender     1
11798411010     KitchenAid Stand Mixer  2

There are 7 items found in one wish list but not the other
Merged wish lists:
11039926010     Digital Kitchen Scale   3
11781701910     Cast Iron Round Griddle 2
11798009510     Espresso Machine        1
11798112010     NutriBullet Blender     1
11798411010     KitchenAid Stand Mixer  2
11798411010     KitchenAid Stand Mixer  1
11881701910     Rice Cooker & Steamer   2
24179114710     Autumn Plaid Tablecloth 2
96796133410     Tan Cotton Blanket      2
```

NOTE: Your name and student ID should be included in the Javadoc comments for your completed WishList class (include an @author tag with your name and student number).

---

2. Testing

Using the test driver provided (do not make any changes to the driver), test your completed **WishList.java** program with 5 different test cases (i.e. run the **TestWishList** class 5 times, with different input each time.) Choose test cases that give you good coverage (i.e. test different scenarios). The example that is provided above may be used as one of the 5 test cases.

For your convenience, the sample input data that is shown on the last page is included in a file named **TestCase1.dat**. This is just a plain text file (so you can open it with a text editor to take a look).

Recall: you can use the redirection operator **<** to send input data to your program from a text file. So, you may find it convenient to create text files to hold the input data for each of your test cases (rather than having to type is all in over and over again each time you run your program).

Similarly, you can use the other redirection operator **>** to send output to a text file from your running program.

When you copy and paste your test input and corresponding results (output) into your report, be sure to explain why each test case that you chose to include is important.

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

i. a written report. This should begin with a title page that includes: the course (CS 1083), the assignment number, your full name, and your UNB student number. That should be followed by two sections, with each part clearly identified with a section heading. Include:
   a. the completed source code for the WishList class (`WishList.java`),
   b. input and associated output for all 5 test cases,
   c. an explanation of each test case.

   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 input and 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 submission folder 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: name this report as follows: **YourName_CS1083_As2_Report.pdf**

ii. an archive file (**.zip**) that contains your Java source code and the files for the 5 test cases 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: name this archive file as follows: **YourName_CS1083_As2_Archive.zip**