G51OOP Coursework 2 - Sets

Question

Given two arrays of integers, you have to perform some array manipulations and write a sorting algorithm. You then have to print the values that occur in each individual array, those which are in both arrays (intersection), the count of the common numbers, those which occur in either array (union), and finally the non-common data for each array (See example output). You will also be required to sort the arrays. For this exercise you must **not** use ArrayLists, any container classes other than arrays, nor any library for sorting; it is expected that you will create the sorting method and write all the code yourself.

Method

Create two empty integer arrays that will hold up to 100 integers. Then, repeatedly read integer values from the keyboard until the user types "0" (zero, without the quotes) and store those values in the first array. Ignore repeated occurrences of a particular value in the data, so check each value to see whether it already occurs in the array before entering it.

Now the array should contain a set (which could be empty) of distinct integer values. Then perform the same procedure to fill the second array.

Afterwards, loop through both arrays and print out, in sorted order from smallest to largest:

a) Raw Data for the first array.
b) Sorted Data for first array.
c) Raw Data for second array.
d) Sorted Data for second array.
e) Common data (intersection).
f) Count of common data.
g) Unique data (union).
h) Non-common data for first array.
i) Non-common data for second array.

See "Typical input" and "Corresponding Output" and adhere to the format specified. Always exclude the terminating zero from your calculations.

Error Conditions

Any of the arrays could be empty, and this should **not** cause your program to behave erratically.

If both arrays are empty, print out "Both arrays are empty", do **not** perform any calculations. If the array is filled (100 entries are made), you should automatically move to the next stage of either filling the second array, or printing out statistics.

Any integer that is entered, whether negative or positive, should be taken into account. Remember that "0" should be excluded from your arrays as this value is meant to stop the input.

Hints

You should break down your program into methods to make your life easier. For example, you may create an "intersection" method that accepts two integer arrays and returns an array of the common values. E.g.

    int [] intersection(int [] firstArray, int [] secondArray)

Other methods might include:

displayElements: accepts an array and prints its contents

fillArray: prompts the user for input and stores the integers into an array

sortArray: takes an array as an argument and returns a sorted version

etc. You should consider the complexity of each method that you make and the affect it has on the logic, readability and maintainability of the code. A suitable combination of calls to your methods should allow easy production of the required output. You are expected to make use of the G51OOPInput.java class.

Notes

You must not use packages for this coursework.

You have one (1) submission. Requests for re-submission with reason stated need to be addressed to Colin Higgins.

Any constant values should be declared as `final` and `static` in your program.

You can be up to two days late with standard University penalties applied (ie 5% per day or part thereof). Later than 2 days will attract a 100% penalty (ie no marks will be awarded).

Typical Input/Output

```
Enter data for array 1 (0 to finish): 1
Enter data for array 1 (0 to finish): 4
Enter data for array 1 (0 to finish): 2
Enter data for array 1 (0 to finish): 5
Enter data for array 1 (0 to finish): 7
Enter data for array 1 (0 to finish): 4
Enter data for array 1 (0 to finish): 8
Enter data for array 1 (0 to finish): 6
Enter data for array 1 (0 to finish): 0
Enter data for array 2 (0 to finish): 5
Enter data for array 2 (0 to finish): 7
Enter data for array 2 (0 to finish): 3
Enter data for array 2 (0 to finish): 11
Enter data for array 2 (0 to finish): 4
Enter data for array 2 (0 to finish): 7
Enter data for array 2 (0 to finish): 3
Enter data for array 2 (0 to finish): 9
Enter data for array 2 (0 to finish): 0

Raw Data for first array is: 1 4 2 5 7 8 6
Sorted Data for first array is: 1 2 4 5 6 7 8
Raw Data for second array is: 5 7 3 11 4 9
Sorted Data for second array is: 3 4 5 7 9 11
Common data (intersection) is: 4 5 7
Count of common data is: 3
Total unique data (union) is: 1 2 3 4 5 6 7 8 9 11
Non-common data for first array is: 1 2 6 8
Non-common data for second array is: 3 9 11
```

Mark Scheme

You will be marked on correctness (if your output is correct for the given input). There are also marks for variable naming, layout, efficiency and output format. Marks will be given for correct use of final variables, selection statements, correct output streams and a sensible choice of loop. The complexity and structure of the methods you use will be assessed. Efficiency of your algorithms will also be marked. As with any programming, you should use comments where necessary in your code to make it more readable. You may be marked down for failure to submit the correct files, or follow the specification (See submission).

Submission

Submission is via Moodle. You should submit a .zip file containing only Union.java and G51OOPInput.java. You should not submit any other format other than .zip (no .iso, .rar, .7z or other alternatives). Make sure to submit the .java files, and not the .class files.