UNBC Laboratory Assignment
Computer Science 200—Fall 2022

Set Operations on Sorted Lists

Purpose:

To practise coding variants of the merge algorithm in a generic context.

Due Date:

This assignment is due **04 November 2022**.

Goal

Your goal is to implement algorithms to compute the union, symmetric difference, difference, and intersection of two sets represented as sorted `ArrayList<Elt>`s.

Background

The `java.util.Collection<E>` related classes support sets through the interfaces `java.util.Set<E>` and `java.util.SortedSet<E>`, and provide a tree-based implementation of sets through `java.util.TreeSet<E>`.

Curiously, none of these classes have operations named `union`, `difference`, `intersection`, *et cetera*. In most cases, these operations exist, but under another name. For instance, (destructive) union is called `addAll`.

Implementing operations like union, difference, intersection, and symmetric difference in $O(n_1 + n_2)$ time, is actually easier with sorted lists than with trees. In each case, the algorithm is a variation on `merge` from merge sort.

Coding Assignment

The actual coding assignment is to implement `union`, `difference`, `intersection`, and `symmetricDifference` with operation with static functions with signatures shown in Figure 1 on the following page. Each algorithm should explicitly loop through the elements like merge, and produce a newly allocated `ArrayList` as a result.

```
public static <Elt> ArrayList<Elt> union
    (ArrayList<Elt> set1,
     ArrayList<Elt> set2,
     Comparator<Elt> c)
```

```
public static <Elt> ArrayList<Elt> difference
    (ArrayList<Elt> set1,
     ArrayList<Elt> set2,
     Comparator<Elt> c)
```

```
public static <Elt> ArrayList<Elt> intersection
    (ArrayList<Elt> set1,
     ArrayList<Elt> set2,
     Comparator<Elt> c)
```

```
public static <Elt> ArrayList<Elt> symmetricDifference
    (ArrayList<Elt> set1,
     ArrayList<Elt> set2,
     Comparator<Elt> c)
```

Figure 1: set operation signatures

Testing

There are two files—`6digits-A.txt`, and `6digits-B.txt`—on the `learn.unbc.ca` website. Each file contains a sorted list of 6 digit numbers.

Write a program that constructs an `ArrayList<Integer>` from each file, and then compute

E1. the union of the two lists (contains 943 numbers)

E2. the intersection of the two lists (contains 587 numbers)

E3. the list of numbers in `6digits-A.txt`, but not in `6digits-B.txt` (contains 137 numbers)

E4. the list of numbers in `6digits-A.txt`, or in `6digits-B.txt` but not both (contains 357 numbers)

Write the results to text files, one number per line, same format as the input files:

F1. write the union to `union.txt`
F2. write the intersection to `intersection.txt`
F3. write the one-sided differece to `difference.txt`
F4. write the symmetric differece to `symDiff.txt`

Research

Search the Oracle Java documentation, and find the names of standard methods acting on `Sets`, that loosely correspond to `union` and `difference`. The standard Java operations are

non-static methods that destructively modify one of the sets.

Write down the names of the methods that you find, and give the URL where you found the documentation.

Checklist

Hand in:

- A `.java` file that implements the four methods shown in Figure 1 on the previous page;
- A `.java` main (test) file that tests the four methods as described in the **Testing** section.
- Four text files containing the answers (list of numbers) that you computed in E1–E4.
- A plain text file that contains your answer to the research question.