CSCI 3353 Object Oriented Design
Homework Assignment 6
Due Monday, March 19

This assignment asks you to write and use iterators.

1. Suppose you have two iterators *i1* and *i2* that are sorted according to a comparator *comp*. Write a class *MergedIterator* that has the following behavior.

```
Comparator<String> comp = ...
Iterator<String> i1 = ...
Iterator<String> i2 = ...
Iterator<String> i3 = new MergedIterator(i1, i2, comp);
```

Iterator *i3* will return the combined elements of *i1* and *i2* in the order determined by the comparator.

The idea is that the merged iterator *i3* will keep a variable that holds the "current" value of each of its component iterators. Its *next* method chooses the smaller of these current values, and replaces it with the next element from that iterator. A current value of null can indicate that its corresponding iterator is done.

Note that this strategy does not pre-process the component iterators. For example, an easy way to implement *MergedIterator* is to have its constructor read all of i1 and i2, saving each value in a new sorted list. Pre-processing such as this is NOT ALLOWED.

2. Write a class *MergedCollection* that has the following behavior.

```
Comparator<String> comp = ...
Collection<String> c1 = ...
Collection<String> c2 = ...
Collection<String> c3 = new MergedCollection(c1, c2, comp);
```

MergedCollection should be a subclass of *AbstractCollection*. You will need to use your *MergedIterator* class from problem 1 to implement its *iterator* method.

3. Download the file *HW6Test.java*. This is a barebones test program that you can use to test your code. When you are convinced that your code works, modify *HW6Test* as follows.

a) Rewrite lines 15-16 of the code so that it uses the *forEach* method instead of a for-each loop.

b) Add code to perform the following test of your *MergedCollection* class: Create two collections, each containing 10 randomly-generated integers between 0-100. Store them in a tree set, using a comparator that sorts them in descending order. Then create a merged collection for them. Print the values in this collection twice: first using a for-each loop, and then using the *forEach* method.

c) Write a method *howMany*, that returns the number of elements in a specified iterable that satisfies a specified predicate. For example:

```
Collection<Integer> c = ...
Predicate<Integer> pred = ...
int x = howMany(c, pred);
```

The header of the method should look like this:

```
public static <T> int howMany(Iterable<T> c,
                              Predicate<T> pred) {
```

You should implement *howMany* using a for-each loop. Then test your method by adding code to *main* that calls *howMany* to determine how many even numbers are in your merged collection of integers. Print the answer.

d) Download my interface *Reducer.java*. Write a reducer class, named *CountReducer<T>*, that counts the number of visited elements that satisfy a predicate; the predicate should be specified in the reducer's constructor. Then write some code in *HW6Test.main* that uses the reducer (and the collection's *forEach* method) to print how many even numbers are in your merged collection of integers.

When you are done, your *HW6Test* program will have printed five lines, containing the following things:

- The sorted strings, as given in the version of *HW6Test* that you downloaded.
- 30 random integers in descending sorted order, printed twice.
- A line saying how many even numbers are in that collection, printed twice.

Here is what my output looks like. Yours should be similar, albeit with different random numbers.

```
a b c d e f g h j m n p
94 93 92 81 79 77 74 74 65 58 55 51 40 38 32 31 28 25 16 2
94 93 92 81 79 77 74 74 65 58 55 51 40 38 32 31 28 25 16 2
There are 11 even values.
There are 11 even values.
```

Finally, create a zip file containing the files *MergedIterator.java*, *MergedCollection.java*, *CountReducer.*java, *Reducer.*java, and *HW6Test.java*, and submit it to Canvas.