Computer Science 501
Data Structures & Algorithms
The College of Saint Rose
Fall 2013

Lab 5: Sorting With Comparators

Due: 6:00 PM, Wednesday, October 23, 2013

This week's lab is designed to introduce (or reintroduce) the power of Java's `Comparator` interface as it applies to writing a generalized sorting method. You will extend the functionality of an existing class using inheritance, you will implement a simple sorting procedure within this extension, and you will learn about `Comparator`s, which provide a more flexible mechanism for ordering objects than the `Comparable`s we have seen in class.

You may work alone or with a partner on this lab. Of course, collaboration with your partner is unrestricted. You may discuss the lab with your classmates and give and receive some help, but your submission must be your own work (or that of you and your partner, if you choose to pair up with a classmate).

Lab Program

Do the laboratory at the end of Chapter 6 in Bailey.

Please note the following clarifications, modifications, and explanations relating to the lab procedure outlined in the text:

- In step 1, you are asked to create an extension of `structure.Vector` called `MyVector`. Since we are using the generic version of the structure package, the class header for `MyVector` should look something like this:

  ```
  public class MyVector<T> extends structure5.Vector<T>
  ```

  Keep in mind that as an extension of `structure5.Vector`, methods of `MyVector` will have access to instance variables and methods declared as `protected` in the `structure5.Vector` implementation. Make good use of this fact!

  Important Note: The `elementData` array in `structure5.Vector` is declared as `private` rather than `protected` for type safety reasons. This means, unfortunately, that `MyVector` will not be able to access the array directly. Fortunately, `Vector` has methods that are almost as good as direct access to the array.

- In step 2, you are to write a `sort` method. The text says you may use any sort that you like, but to earn full credit you should use either merge sort or quicksort for your final version. (Use something simpler for testing if you wish and the use of a simpler sort in your final submission will result in only a small penalty.) The structure of the code will be very similar

to what you have seen in our class and text examples, but you will need to modify it to use `Comparator`s instead of base types or `Comparable`s and to operate on the contents of your instance of `MyVector`.

- Be sure to test your `sort` method in `MyVector` thoroughly before going on to part 3 of the lab assignment.
- For part 3 of the lab procedure, write two applications. They should each work on a different data file and each should perform more than one "interesting" sort process. Both will use the same `MyVector` class, but each application will call its own class for encapsulating the data objects you are working with, and a number of `Comparator`s to sort the data in different ways. You should have a total of at least 5 unique `Comparator`s in your submission.

  You may choose from the data files I have provided in the "`labs/comparators`" directory in the class shared area. See the `README` file there for more information. You may also use some other data that you find interesting.

Thought Questions

Answer the thought questions at the end of the lab. Include your responses in the `README` file described below.

Submitting Your Work

When you're finished, create a tar or zip file (or some similar archiving format) that includes the following:

- Your well-documented source code for all Java files used, including `MyVector.java`, your classes that hold the data objects decide to work with, your `Comparator` classes, and your two applications.
- Any additional data files you choose to use, other than the ones provided.
- A `README` (plain text) file that describes what is in each Java file, describes the two applications you have chosen and what you are sorting and how for each example. Also include your answers to the two thought questions in the text in this `README` file.

Before 6:00 PM, Wednesday, October 23, 2013, submit your Java program for grading. There are two things you need to do to complete the submission: (i) upload a copy of your tar or zip file (for your program, please include `.java` files only, no `.class` files) using Submission Box at `http://sb.teresco.org` under assignment "Comparators", and (ii) print and turn in a hard copy of your program.

Don't forget to check your programs for compliance with the Style Guide for CSC 501 Programs

---

Grading

Grading Breakdown	
Program design	2 points
Program style	3 points
Program documentation	5 points
Program correctness: `MyVector`	6 points
Program correctness: `Comparator`s	5 points
Program correctness: sorting applications	3 points
“Interestingness” of sorting applications	2 points
Thought questions	4 points