Assignment 4: Performance Measurement

Assigned: October 24, 2013
Value: 20 points
Program Due: a JAVA program is due no later than 11:59 pm, Sunday, Nov. 3rd

This is an individual assignment.

Requirements

Experiment Set up - You will create an ArrayList of 1 million random positive integer values using a Random( ) object. Use the default seed for experiment repeatability. Each value is added to the end of the ArrayList when generated.

Open up the "input.txt" file on disk, which contains a sequence of integer search values that you will use for this whole set of experiments (one value per line). All outputs from the experiments will go to the console.

Experiment 1 - You will process each value from the input file in turn, search the ArrayList sequentially, and then report the index where the value was found, or report that it was not found. Do this for all values in the disk file.

Using a Stopwatch object (provided class on BB) you are to measure and report to the total time spent on searching for these values (not including the I/O time to read or report). Close the disk file when done.

Experiment 2 - Next convert your ArrayList into a LinkedList in the same order; you may use the JCF LinkedList class for this purpose. Reopen the disk file and perform the series of searches for values as before. Measure, record and report the time search time (again without I/O time). Close the disk file.

Experiment 3 - Next sort the ArrayList into ascending numerical order. You may use one of the sort methods in the Java API. Measure and report the total time taken to sort the ArrayList into order.

Experiment 4 - Next reopen the file on disk, and again process each value in turn, searching the ordered ArrayList using the binarySearch method from the API. Measure and report the total time spent on searching for these values (not including I/O time to read or report). Close the disk file.

Experiment 5 - You will write your own Interpolation search method for the next phase of the experiment. See http://en.wikipedia.org/wiki/Interpolation_search, and the slides presented in class. Reopen the disk file. Process each value in turn, searching the ordered ArrayList for it using your own interpolation search method. Measure and report the total time spent on searching for these values (again not including I/O time).

Functional testing strategy: Choose a representative set of search values to test with. These should include some low values, some high values, some mid-range values – some of which are found and some which are not found. Also test the upper and lower bounds of the legal value set. Test for catching illegal values as well (e.g. negative numbers) and non-integer inputs too. A total of about 1000 values in the disk file should be sufficient to observe real timing differences.