CSCI 4041 – Algorithms and Data Structures

Programming Assignment 1: Sorting Algorithms: Implementation and Analysis

Released on: Mon., February 15, 2010 -- Due Date: Weds., March 31 2010

Problem Statement:

In this assignment, you are required to implement the following four sorting algorithms in Java, C, or C++: ***Insertion*** sort, ***Merge*** sort, ***Heap*** sort, and ***Quick*** sort. The target platform should be UNIX: Linux (preferred) or Solaris. Please use the GNU compilers for C and C++ (gcc and g++, respectively) and Sun's compiler for Java (javac). Your final executable should be a command-line application called "MySort" that takes two arguments:

1. A letter to indicate the used sorting algorithm: 'i' for insertion, 'm' for merge, 'h' for heap, or 'q' for quick
2. The name of the file to be sorted.

Example: `% MySort i ds_random.txt` would sort the file "ds_random.txt" using insertion sort.

Sample Input:

Your sorting algorithms are to sort, in ascending order, a dataset of 100,000 integers. We have made five different datasets available for download at:
http://www-users.itlabs.umn.edu/classes/Spring-2010/csci4041/files/csci4041data.zip

Filename	Description
`ds_sorted_asc.txt`	This file is already sorted in ascending order
`ds_sorted_dec.txt`	This file is sorted in descending order
`ds_most_sorted_asc.txt`	This file is mostly sorted in ascending order, with 1000 numbers out of place.
`ds_most_sorted_dec.txt`	This file is mostly reverse sorted in descending order, with 1000 numbers out of place.
`ds_random.txt`	This file is not sorted at all.

Submission Instructions:

Please submit a zip or tar.gz file using the electronic submit tool (http://wwws.cs.umn.edu/submit). This archive file should unzip to be a single directory containing the following:

1) A Makefile with targets *compile*, *insertion, merge, heap,* and *quick.* For the sort targets, the Makefile should have several commands, one running that sort algorithm on each data file
2) A README, if necessary, to explain your code
3) A sub-directory named *src* that contains your source code.
4) Please *do not* submit the data files with your assignment. We already have them.

Before you submit, you should double-check that you have correctly organized your assignment. You can do so in the following way:

1) Copy your zip/tar.gz file to an empty directory
2) Unzip the file using unzip or tar xzvf
3) Type 'cd *' into the console
4) Type 'make compile'
5) Type 'make insertion'

You should see the output of your insertion sort program running on all 5 data files. If this procedure does not work, do not submit until you have fixed it. **If you do not follow the submission directions exactly, you will not receive credit for your assignment.** (It is your responsibility to see the TAs immediately if you do not understand any part of these instructions.) The course website will have information on using 'make' and preparing your assignment for submission shortly.

Output:
We are interested in three outputs for each sort/data file combination:

1. The sorted file of the input data stored in a file. The output file name should be the same as the input filename with the suffix ".sorted" appended (e.g., the sorted output of the random input file *ds_random.txt* must be in a file called *ds_random.txt.sorted*).
2. The number of comparisons your algorithm needed to make. This can be done by increasing a counter with each comparison you encounter, and should be written to standard output (the console).
3. The time it took to run the sort. On UNIX machines, you should use the time command to get this in milliseconds. For details on the time command, check the man page at the terminal: `% man time`

Analysis Report:
Once you have recorded the number of comparisons and execution times for each of the four algorithms against each of the five dataset files, you need to represent and analyze your results in a project report. Your report should include at least a graph comparing sorting run time on each dataset, and table(s) with both comparison count and run time for each sort/dataset pair. You should then analyze your results and make conclusions as to the efficiency of the algorithms against different datasets using the metrics you gathered.

Deliverables:
For this assignment there are two deliverables:

1. (80 points) The zipped/tar.gz'd directory containing the items described above. Submit this via the submit tool: http://wwws.cs.umn.edu/submit
2. (20 points) Your report. Please submit ***hard copies*** of this in class on the date due.

Questions:
Any questions you have should be directed at either TA Tim (tim.miller@gmail.com) or Prof. Mokbel (mokbel@cs.umn.edu).