Project 0: Warm-up Programming for Sorting

CSCI 2720 Data Structures (Spring 2009)
Due January 27, 2009

This programming project is to implement two sorting algorithms that you are familiar with, INSERTIONSORT and MERGESORT, and to analyze the running times of the implemented programs through instrumentation. Call these two programs InsertionSort and MergeSort. The project has the following requirements.

1 Correctness

Your implementation of these two programs should be correct. The programs should accept a list of non-negative integers as an input and outputs the sorted list (in the non-decreasing order, for example).

2 Languages

You may use either C++ or Java language for the implementation. Make sure your programs contain sufficient documentations and indentations for format.

3 Instrumentation

In each of the programs, you will embed some instructions to count the total number comparisons used by the program. In particular, wherever a comparison occurs between two elements of the input list, you need to add an instruction to increment the count (e.g., count++). The count should be properly initialized and outputted.

4 Testing

You will write another program, called DataGenerator, to generate data needed for testing the two sorting programs. This program will read in an integer \( N \) and outputs a list of \( N \) non-negative integers. \( N \) can be any non-negative integers; but you may assume that \( N \) cannot
be larger than a million. Generation of the list can be done by using a pseudo-random number

generator built in the programming language you use.

For each of the two programs InsertionSort and MergeSort, test them on data lists
generated by DataGenerator for the following values of N:

\[1,000, 2,000, \ldots 10,000, 11,000, \ldots, 100,000\]

For each of such values of N, you need to run 50 times program DataGenerator to generate
50 different lists. (Note that sometimes a pseudo-random number generator generates the same
list in every rerun. So make sure to adjust a so-called ‘random seed’ to generate different lists).
Then test these lists on InsertionSort and MergeSort to find out the average count of element
comparisons used by each of the sorting programs. The average is taken by dividing the sum
of the total counts obtained from the 50 runs by 50.

5 Analyses

Based on the instrumentation results for each of the sorting programs, use Microsoft Excel to
plot (i.e., draw a curve) for the average counts on \( N = 1,000, 2,000, \ldots 10,000, 11,000, \ldots, 100,000 \)

At the same time, you will use the computer to draw a theoretical running time curve. For InsertionSort, the theoretical curve is \( T_{ins}(N) = cN^2 \), where \( c \) is a constant you would
like to find out. To draw this theoretical curve, you will need to generate a list of values \( cN^2 \),
using \( c = 1 \) for \( N = 1,000, 2,000, \ldots 10,000, 11,000, \ldots, 100,000 \), and use Excel to plot these
data. Then you try to correlate the two curves, one theoretical and one obtained from the
instrumentation, by adjusting the value for \( c \) so that the two curves are ‘close enough’. Identify
the value of \( c \) you find.

For MergeSort, the analysis process is essentially the same but you will use the function
\( T_{mer}(N) = cN \log_2 N \) for the theoretical curve instead.

6 Submission

You need to submit the three programs InsertionSort, MergeSort and DataGenerator elec-
tronically to the CSCI 2720 account. You also need to submit a report to the instructor paper
copies of these programs together with your analyses (curves and \( c \) values) for both sorting
programs.