# Purpose

The purpose of this assignment is to learn to analyze algorithms using a profiling tool and report on your conclusions.

# Specifications

1. Write a short program that fills different **list<int>’s** of 100 integers with the same **random non-negative integer values**, sorts them using the **Sink, Selection, Insertion,** **Mergesort**, **Quicksort (original), Quicksort (median-of-three), Shell (gap** determined using **2.2** as in the lecture**)**, **Counting**, and **Radix (base 10, least significant digit)** routines. The basic code for implementing these algorithms is given in the lecture material.

In successive executions of the program, repeat the process above with data that has the following characteristics (4 total original orderings).

* Already in order
* In reverse order
* Almost random data – about a 10% not in order.

1. Now repeat the entire process with lists of 1000 integers and then with lists of 10,000 integers. Because of the recursive nature of some of the routines and the expanding list and profile sizes, you may reach a point where increasing N causes an error due to resource limitations. If so, don’t try to resolve the problem – just stop there.
2. For each run of the program, **profile** the execution (using ***instrumentation*** – not ***sampling***), paying particular attention to the performance of the sort routines. Record the profiles in tabular format in Excel to make it easy to compare the profiles among the routines, among the different values of N, and among the different data set characteristics. Every table, row, and column must be labeled with a meaningful label. Use Excel to produce a meaningful **graph** showing comparisons of the algorithms for each data set.
3. In analyzing the data, you are looking for patterns so that you can draw conclusions **such as** those illustrated by the following statements. The phrases in this list **are not** the particular observations that you are expected to find, but they should give you suggestions as to the types of patterns you are seeking. If you think you see a pattern, you may want to test it with other data to verify the conclusions. The following are **just examples.**

* The difference between the run time for the **Insertion** and **Sinking** sorts < .0001 second in all cases
* The **Quick** **Sort** is always faster than the **Selection** **Sort** on random data but the **Selection** **Sort** is faster on data that is already sorted
* If the number of entries goes up by a factor of 10, the **Quick** **Sort** takes 102 times as long
* The **Insertion** **Sort** is useful for data sets that are almost ordered but is not good for random data
* The **Sinking** **Sort** works well for small data sets but not for data sets of 1000 or more items

1. Look for patterns in the results. Look for patterns involving:
   1. the same algorithm with different values of **N**
   2. different algorithms with the same data sets
   3. the same algorithm with data sets that have different characteristics but the same **N**.
2. You are seeking answers to such questions as the following:
3. Is there a pattern that would cause me to choose sorting algorithm X over the others?
4. Is there an algorithm that is always best regardless of the data characteristics?
5. Is there a set of data characteristics for which I should always avoid using algorithm Y?
6. Is there a set of algorithms that are roughly equal in performance so that it doesn’t make much difference which I choose from that set?
7. Is there an algorithm that I should never use regardless of the data characteristics when there are more than (fill in the blank) items to sort?
8. Are there any obvious improvements I can make to any of the algorithms, and what evidence do I have for the expected improvement?
9. Your test program should be very simple. In order to compare the profiles of the algorithms, you should run each test on one set of data, record the profile information, and then modify the program slightly (say, increase **N** by a factor of 10 or change the data from random to already sorted) and run it again. You should verify that your sorting algorithms produce correct results, but when you run the profiler you should remove or comment out all I/O statements.

# Deliverables

Turn in a copy of the program code for one iteration (**do not submit** the **profile files**) and the spreadsheet(s). Write an MS-Word report that includes your observations and conclusions. Include your well-labeled spreadsheet tables and labeled graphs as appendices in the report. The data in the spreadsheet tables must support your conclusions.

The report should also state theoretical findings such as a **big-Oh** analysis ***if and*** ***where you can***. A **big-Oh analysis** is not required for all algorithms. Does your data confirm the theoretical analysis? If not, why? The spreadsheets and the report should include clearly labeled graphs that illustrate your points.

The report should have the standard sections for topics such as:

* introduction
* problem statement
* what you did and observations you made based on the data
* conclusions that you reached
* summary

It should have a cover (title) page and use footnotes to associate your stated observations with the basis for them in the data.

Use stylistic tools such as section headers to organize your report and make it both attractive and easy to read.

Use tools that are commonly available to you in Word such as spell checkers, grammar checkers, and so forth to improve the quality of the writing in your report. [The accompanying skeleton report shows the expected structure of the report.](http://csciwww.etsu.edu/bailes/courses/2210/assignments/Profiling%20Research%20Paper%20Skeleton.docx)

Please use an **.xls** or **.xlsx** extension (format) for the spreadsheets and a .**docx** extension/format for the **Word** document (that is, don’t expect the grader and me to have every version of every open source or bootlegged word processor and spreadsheet program there is – all computers in the labs have current versions of both **Word** and **Excel – and current versions are free from Microsoft for student use**). Submit everything as part of a zipped file with an appropriate name using the conventions described in the **Course Facts** sheet on the web.

There is no minimum length for the report, but using extra words and huge fonts just to make the report longer is not appropriate and will diminish your grade. Please use common fonts, font sizes, margins, etc. A rule of thumb is to **make reports as concise and simple as possible**, but not more concise or simpler than that.

Your grade will be based on the data you collect, the format in which you present it, the analysis, and conclusions in your paper, and on the quality of the writing. **Spelling, grammar, report structure, and so forth count**! Be as concise as possible, but provide convincing evidence for your conclusions.

**You should take advantage of the proofreading assistance in the ETSU and/or College of Business and Technology writing labs to help you achieve the best grade**.