**Project 2**

**Sorting, CPU Time, and Big O Analysis**

This is an individual assignment. You will have lab time to work on this assignment as well as homework time.

Each person will analyze two sorts. The first sort will be either bubble sort or insertion sort. The second sort will be either quick sort or merge sort. Create two projects, one for each sort in the given choices.

In this assignment, we will be learning about, describing, coding, and empirically looking at the work involved in each sorting algorithm. Use 4 data set sizes. Results will be based on execution time for each sort. The size should have a regular increment. For example, using a regular increment of 10,000, you might have sizes of 10,000, 20,000, 30,000, and 40,000. Choose your data size based upon the ability to obtain meaning results.

Submit:

1. A Word document named SortDescriptionsLastName.docx. The Word document should be modeled after the selection sort description under Course Content. The Word document should have:
   1. A few useful web resources for the sort.
   2. A detailed explanation (in your own words) of the sort.
   3. A clear visual representation of the sort (cite source if found on web, book …).
   4. Two screen shots of your program displaying the sorted array for each sort. The display of each value should include the index and value of each element in the array. These screen shots of course will not show your entire array but only a part.
2. An Excel document that shows the empirical results of running each sort. The name of the file should be SortEmpiricalAnalysesLastName.xlsx.
3. Code for the algorithm in a working program (similar to SelectionSort.cpp).
   1. Code should be documented.
   2. Code should use meaningful identifiers (do not use abbreviated variable names such as ***r*** for ***right***).
   3. Code should be well-structured and easy to read.
   4. You should be able to understand every aspect of the code that you are using.
   5. The code for each sort should be organized in its own project folder – a compressed zipped folder for each project. Use only .zip folders (not .rar folder).

Grading

|  |  |
| --- | --- |
| For the two sorts:  Code is well documented  Code uses meaningful identifiers  Code is well-structured and easy to read.  In Word document:  Useful web resources for each sort  Detailed explanation (in your own words) of each sort  A clear visual representation of each sort (cite source if found on web, book …)  One screen shot of your program displaying the sorted arrays  An Excel spreadsheet that shows the empirical results of running each sort.  Code for each sort should be organized in its own project folder – a compressed zipped folder (for each project). Use only .zip folders (not .rar folder). | 10 points  10 points  10 points  10 points  20 points  10 points  10 points  10 points  10 points  Total 100 |