CSCE 156 Lab: Sorting & Efficiency

Worksheet

Name \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Verify that your sorting algorithms are correctly sorting by printing the content of the arrays and submit your results.
2. Run some timed experiments as outlined in the lab handout for each algorithm for various input sizes. Note that you can restrict the number of locations loaded from the data file by changing the value of n in the main method of the SortingPerformance class. Fill in the table below (for best results, run the experiment at least three times each and take an average running time unless you’re feeling lazy).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Algorithm | Theoretical Efficiency | Observed Average Running Time (seconds) | | | |
|  |  | n = 2000 | n = 4000 | n = 8000 | n = 20000 |
| Java Sort | O(nlog(n)) |  |  |  |  |
| Buble Sort | O(n^2) |  |  |  |  |
| Selection Sort | O(n^2) |  |  |  |  |
| Insertion Sort | O(n^2) |  |  |  |  |
| Quick Sort | O(nlog(n)) |  |  |  |  |

1. Without actually running the simulation, predict the running time of each algorithm for n = 64,000 based on the theoretical efficiency and observed running time.
2. According to your experiments, is there a clear ranking of the sorting algorithms? If so, list them from best to worst.