Rowan Antonuccio

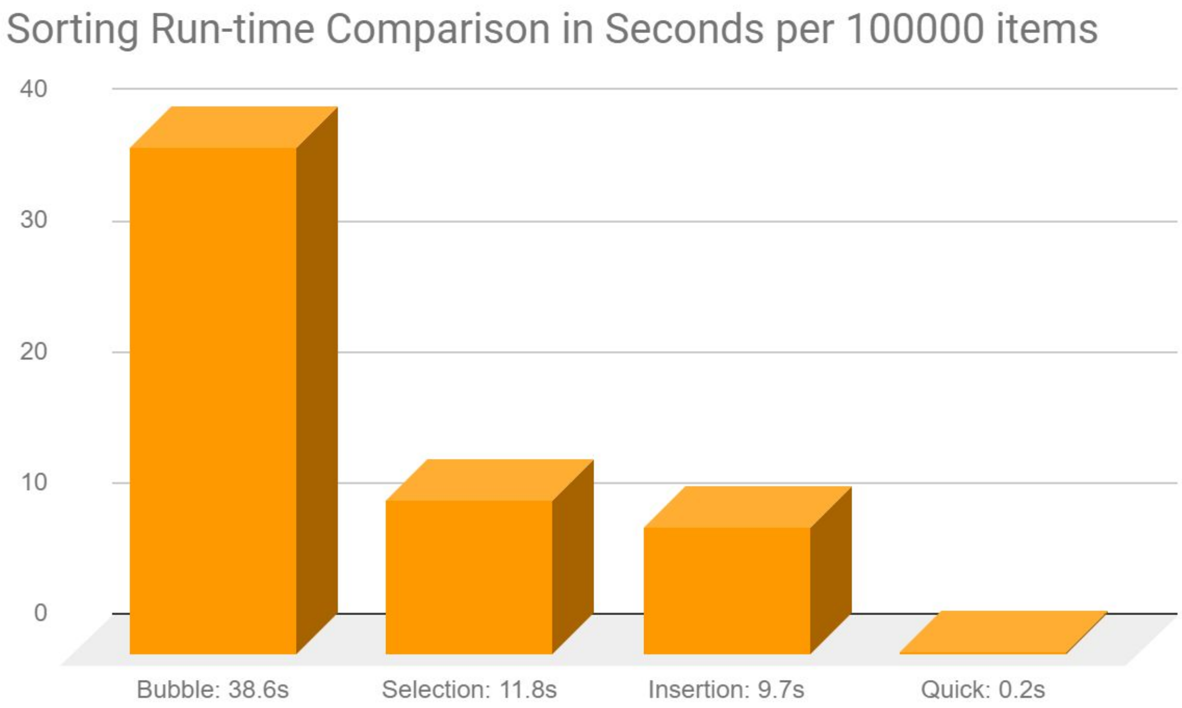
ECE 1410

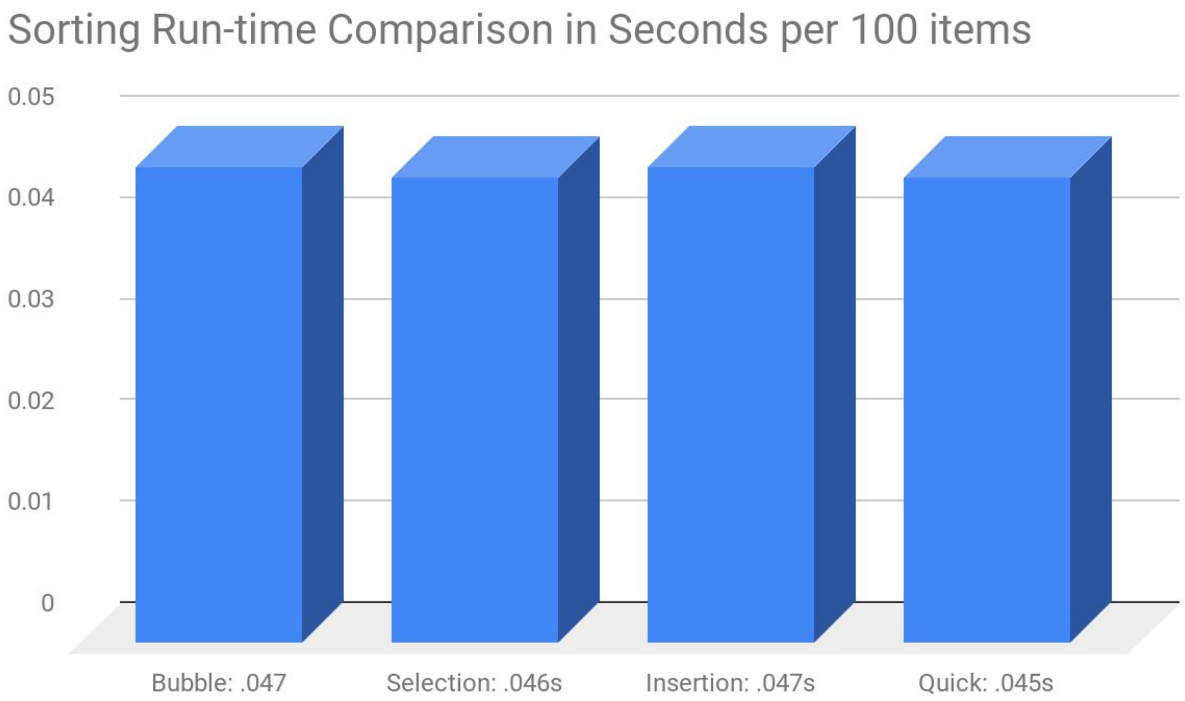
Jonathan Phillips

4/12/2019

Sorting Efficiency

After analyzing the four sorting methods, there seem to be significant benefits to using methods other than bubble sort. While bubble sort is extremely simple, it's execution is very un-optimized. While more complex, selection and insertion are a good middle grounds for sorting methods because they are not as complex as quick sort, but are far faster than bubble. On the other hand, once you've gotten past the complexity of quick sort, it is by far the best method to use if efficiency and run-time are of value to the programmer. If the program at hand was only sorting under 100 items, then any of the above methods are viable. Also, with a list as small as 100, the simplicity of Bubble can outshine the long run time because on short lists it's run time is up to par with the others and the difference would not be noticeable by the user.





As seen in the charts above, the difference in run-time between sorting methods depends entirely on the amount of items being sorted. As shown when run-times for both 100,000 items and 100 items are compared. In the test with 100 items, none of the methods are any noticeable value faster than another, which means that bubble would be the most lightweight and simple method to implement. But on a program sorting 100,000 items, the difference is huge and quick sorting would be by far the best option unless run-times of 10 seconds or more are acceptable. This goes to show that each method has value and can be efficient when implemented in a proper scenario.