Assignment 6

December 13, 2019

Sebastian Ludlow

1 Sort Differences

When comparing the origional numbers in miliseconds for each of the four sorts used, I was surprised to see that the Insertion sort was the most efficent in processing the list of 5000 random numbers. Averaging to be a whole 5-10 ms faster than the second quickest sorting system the Selection sort process. Judging the sorting time in miliseconds helped me appreciate the tiny differences programmers can seek to attain in optimising their code. Especially considering that each sort isn't really too different in the amount of code required to create the sorts yet the run times were wildly varied as a result.

2 Favorite Sort

Perhaps my most favorite sort to use out of the four I experimented with was the gnome sort which functioned similary to the insetion sort. In that it only focused on a single variable but also peformed a more complex sort function of the bubble sort without using a ton of nested for loops in the process which interestingly enough led to a longer complete time which I thought would be the inverse of the result.

3 Problems

While testing the runtimes of the different sorts felt very successful the only thing I would change is perhaps running this in another type of language such as Java to compare the runtime results and see directly across similar sort types if C++ is better at handling the sort methods or vice versa. I also peformed the compile and tests on a lower end pc which I am curious as to if that might have affected the times or if a more poweful desktop would handle it any faster or have no impact at all.