COMP2521 Sort Detective Lab Report

by Prithvi Sajit (z5358751), Jacky Al-Siddiq (z5359103)

In this lab, the aim is to measure the performance of two sorting programs, without access to the code, and determine which sorting algorithm each program uses.

Experimental Design

We measured how each program's execution time varied as the size and initial sortedness of the input varied. We used the following kinds of input; Ascending, Descending, Random, Duplicates only,

We used these test cases because they were able to distinguish between the algorithms

Because of the way timing works on Unix/Linux, it was necessary to repeat the same test multiple times to get a true reflection of the performance. For random inputs, it was necessary to include a seed number in order to have consistent tests across different algorithms.

We also investigated the stability of the sorting programs by having duplicate values in some of our tests

We also investigated time complexity by timing how long tests would take to complete for each algorithm.

Experimental Results

For Program A, we observed that it is not efficient at dealing with large sets of data. We also observed that the input type did not affect the results

These observations indicate that the algorithm underlying the program *has the following characteristics;* is slow for larger data sets, works the same with different input orders.

For Program B, we observed that it is very efficient at dealing with data regardless of the size. We also observed that the input of the tests did notably affect the times. We did notice that the times did fluctuate.

These observations indicate that the algorithm underlying the program *has the following characteristics* is fast for even large data sets

Conclusions

On the basis of our experiments and our analysis above, we believe that

- sortA implements the *Bubble Sort* sorting algorithm
- sortB implements the Randomised *Quick Sort* sorting algorithm

Appendix

Any large tables of data that you want to present ...

SortA n = 16000

	Ascending	Descending	Random	**Best**	**Worst**	**Average**
Time	0.46	0.44	0.46	0.44	0.06	0.45

SortA n = 40000

	Ascending	Descending	Random	**Best**	**Worst**	**Average**
Time	2.85	2.75	2.86	2.75	2.86	2.82

SortA n = 160000

	Ascending	Descending	Random	**Best**	**Worst**	**Average**
Time	45.75	42.95	45.63	42.95	45.75	44.78

SortB n = 16000

	Ascending	Descending	Random	**Best**	**Worst**	**Average**
Time	0.00	0.00	0.00	0.00	0.00	0.00

SortB n = 40000

	Ascending	Descending	Random	**Best**	**Worst**	**Average**
Time	0.00	0.00	0.01	0.00	0.01	0.003

SortB n = 160000

	Ascending	Descending	Random	**Best**	**Worst**	**Average**
Time	0.03	0.03	0.06	0.03	0.06	0.04