CPS 616 - W2017 **Assignment – Profiling Efficiencies** **Due March 17**

PLEASE SUBMIT A PRINTOUT OF THIS SHEET TO THE COMPUTER SCIENCE OFFICE

D2L submission folder under which Java files were submitted electronically: A 33

Name: Aumkar Prajapati **Student ID:** 500563404

Name: Baheer Kamal **Student ID:** 500574632

PAPER SUBMISSION CHECKLIST – Please check all your attachments

- ☑ Questions
- ☑ Summary spreadsheet
- ☑ Plot

MARKING RUBRIC

Part 1 (10 marks)	Slowsort described in a) is implemented properly and works	/ 5
	Fastsort described in b) is implemented properly and works	/ 5
Part 2 (15 marks)	Assignment properly prepared for profiling	/ 5
	Profiling properly conducted and recorded in summary sheet	/ 10
Part 3 (15 marks)	Correct plot	/ 15
Questions (20 marks)	a: (3) b: (3)	/ 6
	c	/ 3
	d: S(4) F(4)	/ 8
	i	/ 3
Late Penalty	March 20: -2 March 21: -5 March 22: -16 March 23: -48 (-60 afterwards)	
TOTAL		/60

Which $O(n^2)$ algorithm are you implementing in this assignment:

☑ Bubble sort ☐ Insertion sort ☐ Selection sort ☐ Other: _____

Which O(n log n) algorithm are you implementing in this assignment:

☐ Mergesort ☑ Quicksort ☐ Heapsort ☐ Other: _____

a. Why are you being asked to graph your values using a logarithmic x-axis? (you can see what the graph would look like otherwise by using the Matlab command *plot* instead of *loglog*).

The x-axis is the number of objects. the plot command doesn't allow the x axis to be scaled correctly as we are working with a very large data set (high objects over high amounts of ns)

b. Why are you being asked to graph your values using a logarithmic y-axis? (you can see what the graph would look like otherwise by using the Matlab command *semilogx* instead of *loglog*).

We use a logarithmic y-axis as we need appropriate scaling that fits all the points & allows them to be distinguishable. The y-axis shows the amount of ns elapsed while the x-axis goes over the n amount of objects.

c. For which values of n does it make more sense to use slowsort instead of fastsort?

n values from 1-17 make more sense w/slowsort instead of fastsort.

d. Slowsort $\in O(n^2)$ and fastsort $\in O(n \log n)$. This means that there are two coefficient S and F such that for all values of n large enough slowsort(n) $\leq S.\ n^2$ and fastsort(n) $\leq F.n \log n$. When you manipulated your graphs, what small values of S and F did you find which meet this criteria?

The values of S & F found were 12 & 2, respectively. This met the criteria.

e. Do you have any comments about the behaviours of your slowsort and fastsort?

To track sort time, we used JDK 8 lambda notation to initiate time tracking. The slowsort is a bit better in the beginning due to overhead, however this changes as n increases