Laboratory 6 – Week 6

## Sorting Algorithms

## 6.1 Introduction

**Firstly, this worksheet *is* one of the worksheets from which your laboratory worksheets portfolio of work will be assessed.**

This laboratory involves the testing of three sorting algorithms. These algorithms are known as SortA, SortB and SortC. You will be required to write a program which tests these algorithms for different sizes of data. You will also be required to create plots of the algorithms performance. Make sure you save any programs or results.

## 6.2 Exercise 1: Preliminaries

Create a project and associated class called Lab6. Extract the embedded object in Appendix B and save it as ThreeSorts.java. Include this java file into your project.

Examine the Java code.

The three sorting algorithms are all of the following form:

res = ThreeSorts.*SortA*(array);

Here array is of type ArrayList<Integer> and is the array to be sorted. Each function returns a sorted ArrayList<Integer> which is assigned to res in the example above.

There are similar methods SortB and SortC for the other two sorting algorithms.

**These algorithms only sort whole numbers (integers) between 0 and 255.**

Do the following:

1. Create a method RandomArray that takes in as parameter an integer n and returns a ArrayList of Integers (ArrayList<Integer>) that consists of n random numbers between 0 and 255 (i.e. the returned array is of size n). Appendix A contains some example code that might be useful.
2. Create a method ShowArray that takes in as a parameter an ArrayList of Integers (ArrayList<Integer>) and displays the contents of the ArrayList.
3. Test these two methods, i.e. create a random list and then display it.
4. Verify with a small sized random array that the three sorting methods do indeed sort an ArrayList.
5. What order do they sort the arrays in?
6. Can you tell from the source code which type of sorting method the three algorithms are?

## 6.2 Exercise 2: Experimental Analysis

The purpose of this exercise is to test the three algorithms on different sized random inputs. The size of each input should range between 1,000 and 25,000 items. Each algorithm should be run at least 10 times on each different size so that the average performance can be measured. Draw a graph of the results; see Appendix C for an example graph. Use your results (and run extra experiments if necessary) to determine the following:

1. Which one is the overall fastest and which one is the slowest algorithm? Rank the algorithms from fastest to slowest.
2. Is this ranking independent of the size of the input? I.e. does it always hold true? Would you use one algorithm for smaller sizes, and another for larger?
3. What is the largest sized array the best method can sort?
4. What is the largest sized array the slowest can sort in a reasonable amount of time?

### 6.3 Appendices

### Appendix A. Random Numbers

The following small section of code will create a random Integer ranging between 0 and 255.

Random rand = **new** Random();

rand.setSeed(System.*currentTimeMillis*());

Integer r = Math.*abs*(rand.nextInt() % 256);

### Appendix B. ThreeSorts Class

The following class contains all of the sorting code you will need for this exercise sheet.

|  |
| --- |
|  |

### Appendix C. Example Convergence Graph

The following is an example performance plot of input size against average run time in nanoseconds for three algorithms X, Y, and Z