Family Name: ..... **Other Names:** .....

ID Number: ..... **Signature** .....

COMP 103: Test 2

7 October, 2013

Instructions

- Time allowed: **45 minutes**
- There are 45 marks in total.
- Attempt **all** of the questions.
- *Read each question carefully before attempting it*.
- Write your answers in the boxes in this test paper and hand in all sheets.
- You may use paper translation dictionaries, and calculators without a full set of alphabet keys.
- You may write working on this paper, but make sure it is clear where your answers are.

Questions	**Marks**	
1. Costs and Sorting	[15]	
2. Trains	[12]	
3. Trees	[18]	
	TOTAL:	

Question 1. Costs and Sorting [15 marks]

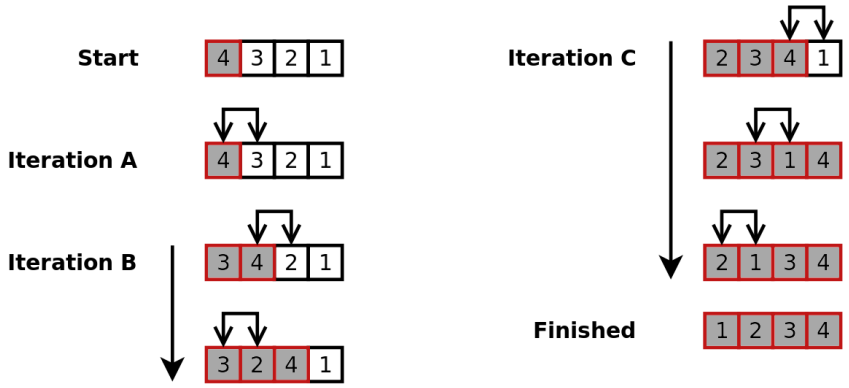
(a) [4 marks] Complete the following table showing the costs of some operations in various implementations of data structures. The first row is done for you.

- for implementations of *List*, the add method adds a new item to the *end* of the list.
- A LinkedSet would be an implementation of *Set* that uses a linked list of nodes to store the items, and so a SortedLinkedSet is the same except that the nodes are kept in *sorted order*. You may assume that the SortedLinkedSet class maintains a single reference, to the head node of the linked list.

	contains(item)	add(item)
ArrayList	$\mathcal{O}(n)$	$\mathcal{O}(1)$
LinkedList		
ArraySet		
SortedArraySet		
SortedLinkedSet		

(b) [3 marks] Which sorting algorithm is the following figure illustrating? Name the algorithm and very briefly explain how you recognised it.

Note: the symbol ↓↓ indicates that a comparison is being made.



(c) [4 marks] In Java, Strings have a method substring(int a, int b), which returns the contents of the string from the character index a up to but not including the index b.
For example `"maybe".substring(0,3)` returns the string `"may"`.

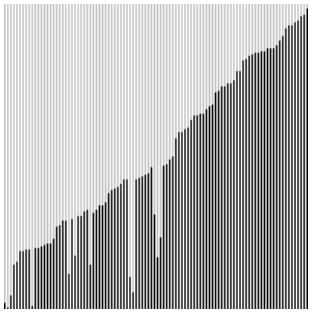
Consider the following recursive method:

```
public void recMeth(String str) {
    if (str.length() >= 1) {
        System.out.println(str);

        int halfway = str.length()/2;

        recMeth(str.substring(0, halfway));
        recMeth(str.substring(halfway, str.length()));
    }
}
```

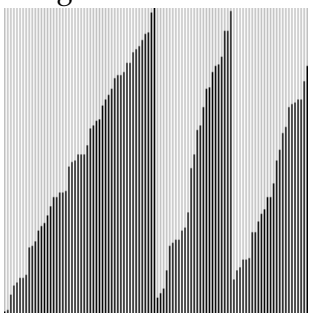
In the box below, show the output produced by recMeth("ABCD");

(d) [4 marks] The following figures show BubbleSort, MergeSort, and QuickSort part of the way through the sorting operation. Each figure shows a snapshot of the whole array being sorted, with the height of the dark bars representing the value in the respective array cell. Which algorithm is closest to finishing? Explain your choice.

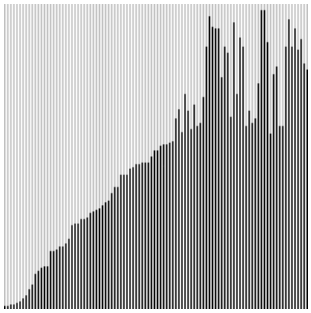
BubbleSort



MergeSort



QuickSort



Algorithm:

Reasoning:

Question 2. Train Simulation [12 marks]

The KiwiRail company needs software to simulate the configuration and dynamic re-configuration of trains. A train has a locomotive, followed by an arbitrary number of carriages. KiwiRail wants you to support the following operations:

OP1 joins two trains by appending the carriages. After the join which preserves the original ordering of the carriages, the locomotive that would end up in the middle is removed.

OP2 splits a train in half, and assigns a new locomotive to the carriages of the second half.

(a) [4 marks] Which standard data structure from the Java library would you choose as a basis to create a data structure that supports the above two operations with the best possible performance? Justify your choice.

(b) [3 marks] In order to achieve the best possible performance for OP1, do you need to extend the basic data structure you chose? If so, briefly explain the principle of your solution. If not, briefly explain why not.

(c) [3 marks] In order to achieve the best possible performance for OP2, do you need to extend the basic data structure you chose? If so, briefly explain the principle of your solution. If not, briefly explain why not.

(d) [2 marks] For both operations OP1 and OP2, state the complexity of the operation using the O-notation, given your solution choices above. Briefly justify each answer.

Question 3. Trees [18 marks]

Consider a general tree for representing the structure of a scientific paper. The nodes of the general tree should contain names for sections, subsections, sub-subsections, etc. Your ultimate goal is to print the contents listing for a scientific paper as in the following example:

```
PaperDraft
  1.Problem
    1.1.Problem Description
    1.2.Problem Analysis
  2.Solution
    2.1.Solution Approach
      2.1.1.Approach Details
```

Consider the following code code.

```
GeneralTreeNode n1 = new GeneralTreeNode("PaperDraft");
GeneralTreeNode n2 = new GeneralTreeNode("Problem");
GeneralTreeNode n3 = new GeneralTreeNode("Problem Description");
GeneralTreeNode n4 = new GeneralTreeNode("Problem Analysis");
GeneralTreeNode n5 = new GeneralTreeNode("Solution");
GeneralTreeNode n6 = new GeneralTreeNode("Solution Approach");
GeneralTreeNode n7 = new GeneralTreeNode("Approach Details");

n1.addChild(n2);

n2.addChild(n3);
n2.addChild(n4);
```

(a) [2 marks] The code above results in several nodes connected in a tree. Draw that tree.

(b) [3 marks] Add statements to the code so that all nodes n1−n7 are part of the tree *and the tree has the structure shown in the contents listing shown at the beginning of the question*.

(c) [3 marks] Assume that method addChild() of class GeneralTreeNode also sets a respective parent reference and that the field name of this reference in class GeneralTreeNode is "parent". Complete the following fullReferencePlain() method of class GeneralTreeNode so that n3.fullReferencePlain() will print "Problem Description in Problem in PaperDraft".

```
public void fullReferencePlain() {

}
```

(d) [4 marks] Complete the following fullReferencePosh() method of class GeneralTreeNode so that n3.fullReferencePosh() will print "Problem Description (within Problem (within PaperDraft))".

```
public void fullReferencePosh() {

}
```

(e) [6 marks] Complete the following printContents (...) method of class GeneralTreeNode so that n1.printContents("") will print the contents listing shown at the beginning of the question. *Make sure to produce the correct indentation and hierarchical numbering*.

To access the name of a node, simply directly access the name field of class GeneralTreeNode.

```
public void printContents(String prefix) {




    int counter = 1;
    for (GeneralTreeNode child : children)
    {









    }
}
```

****