TE WHARE WĀNANGA O TE ŪPOKO O TE IKA A MĀUI
VICTORIA
UNIVERSITY OF WELLINGTON

EXAMINATIONS — 2006

END-YEAR

COMP103

********** **WITH**

SOLUTIONS **********

Introduction to
Data Structures and Algorithms

Time Allowed: 3 Hours

Instructions:

1. Attempt **all** of the questions.

2. *Read each question carefully before attempting it.* (Suggestion: You do not have to answer the questions in the order shown. Do the questions you find easiest first.)

3. This examination will be marked out of **180** marks, so allocate approximately 1 minute per mark.

4. Write your answers in the boxes in this test paper and hand in all sheets.

5. Non-electronic foreign language-English translation dictionaries are permitted.

6. Calculators are permitted.

7. There is documentation on the relevant Java classes and interfaces at the end of the exam paper.

Questions		**Marks**
1.	Basic Questions	[23]
2.	Collections	[15]
3.	Interfaces and Abstract Classes	[10]
4.	Sorting	[23]
5.	Binary Search Trees	[34]
6.	Tree Traversals	[20]
7.	Heaps	[16]
8.	Hashing	[18]
9.	Graphs	[21]

Student ID: .....

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

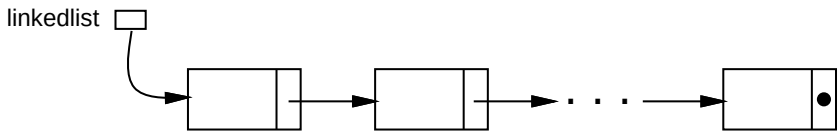
Student ID: .....

Question 1. Basic Questions [23 marks]

(a) [2 marks] State one difference between an Array and an ArrayList.

(b) [2 marks] What happens to an object when it is no longer referenced by any other objects?

(c) [2 marks] In what way is access to data within a Queue constrained?

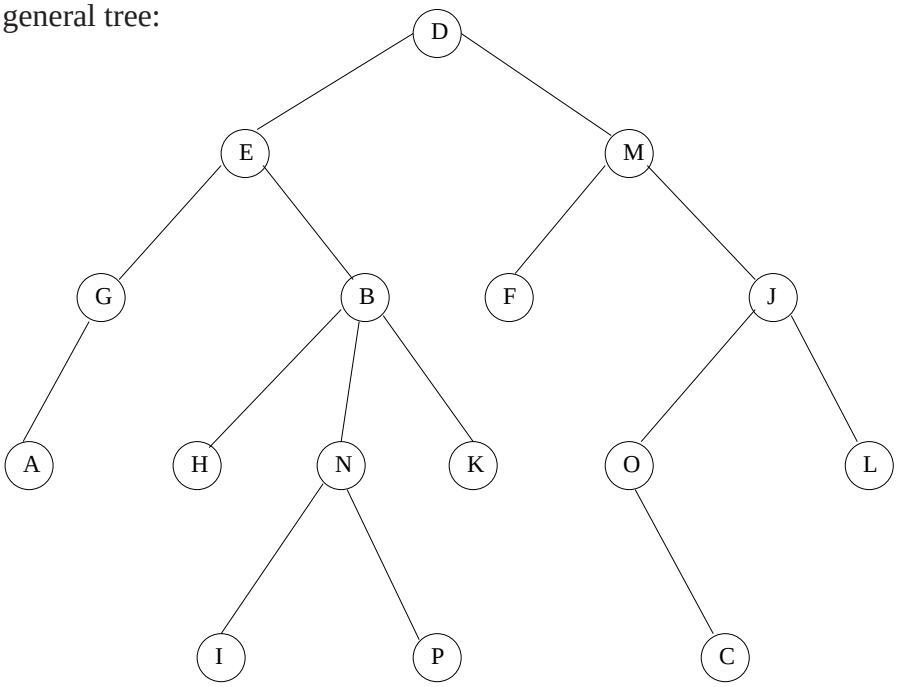


(d) [2 marks] What is the big O cost of inserting an item into the start of the above linked list?

(e) [2 marks] What is the big O cost of inserting an item at the end of the above linked list?

Student ID: .....

Consider the following general tree:



(f) [2 marks] What value is in the parent of the node with the value "O"?

> "J"

(g) [2 marks] How many leaf nodes does the tree have?

> 8

(h) [2 marks] Why is the tree NOT a binary tree?

> Because node B has three children

(Question 1 continued on next page)

Student ID: .....

(Question 1 continued)

(i) [3 marks] Explain the difference between Set, Bag, and Map

(j) [2 marks] What is the average case Big-O cost of adding an item to a set of *n* items implemented using a Hash Table with quadratic probing, if the Hash Table is less than half full.

O(1)

(k) [2 marks] How many levels in a complete binary tree with 16 nodes?

5

Student ID: .....

Question 2. Collections [15 marks]

(a) [10 marks]

Draw lines between things on the left and the collections on the right indicating the best choice of collection for that thing. Note: it may or may not be a 1:1 mapping.

Music notes in a tune.	STACK | List
Clothes in a load of washing.	QUEUE | Set
Layers in sedimentary rock.	SET | Stack
Cars in a drive-through.	MAP | Queue
Student IDs and Names.	LIST | Map

(b) [5 marks] Briefly explain why generic type variables are of great importance for collections in Java.

Because you can define collections that can be instantiated to contain any type, yet remain type safe/consistent.

Student ID: .....

Question 3. Interfaces and Abstraction Classes [10 marks]

(a) [4 marks] State whether each statement is true or false by circling the correct answer.

1. You can instantiate an Abstract Class. True False False
2. You cannot write method code in an interface. True False True
3. An Abstract class can Implement an Interface. True False True
4. An Interface can Extend an Abstract Class. True False False

(b) [6 marks] Consider the following declarations.

```
public interface SomeThing {
    ...
}
public interface SomeThingElse {
    ...
}
public class EvenEarlierThing implements SomeThingElse {
    ...
}
public class EarlierThing {
    ...
}
public class Thing extends EarlierThing implements SomeThingElse{
    ...
}
```

Suppose an instance of the Thing class is assigned to a variable. List all the possible types of the variable.

Thing, EarlierThing, SomeThing and Object

Student ID: .....

Question 4. Sorting [23 marks]

(a) [10 marks] Consider the problem of sorting Cars in an automotive show room. There are many ways to sort the vehicles, price, size, age, colour, etc. The manager wants a program that will workout the layout of his showroom. The Car class is given below.

```
public class Car {
    private String registrationNumber;
    private int yearOfRegistration;
    private int price;

    public String getRego(){
        return registrationNumber;
    }

    public int getYear(){
        return yearOfRegistration;
    }

    public int getPrice(){
        return price;
    }

    public int getAge(){
        return 2006-yearOfRegistration;
    }
}
```

(Question 4 continued on next page)

Student ID: .....

(Question 4 continued)

Instances of the Car class are stored and manipulated by the following Showroom class. Complete the following Showroom class by supplying the comparator classes for ordering by increasing age and price.

```
public class Showroom {
    private Collection inventory;
    public Showroom(){
        inventory = new ArrayList<Car>();
    }
    public void SortByAge(){
        Collections.sort((List)inventory, new AgeComparator());
    }
    public void SortByPrice(){
        Collections.sort((List)inventory, new PriceComparator());
    }
```

```
/* Comparator to order cars by increasing age (most recent first) */
private class AgeComparator implements Comparator<Car> {









}
```

```
/* Comparator to order cars by increasing price (cheapest first) */
private class PriceComparator implements Comparator<Car> {









}
```

(Question 4 continued on next page)

Student ID: .....

(Question 4 continued)

(b) [5 marks] Choose your favourite fast sorting algorithm out of MergeSort, Quicksort, ShellSort, and Radix Sort. Outline the basic idea of the algorithm.

(c) [8 marks] Suppose an ArrayList containing the following values is to be sorted using MergeSort (recursive version).

zucchini	courgette	apple	pear	melon	grape	tamerillo	eggplant
0	1	2	3	4	5	6	7

Show the state of the ArrayList at the end of each of the first three calls to Merge.

after 1:

0	1	2	3	4	5	6	7

after 2:

0	1	2	3	4	5	6	7

after 3:

0	1	2	3	4	5	6	7

Student ID: .....

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Student ID: .....

Question 5. Binary Search Trees [34 marks]

(a) [3 marks] Draw a Binary Search Tree containing the values 14, 79, 86, 5, 4, and 31. The root should contain the value 14.

(b) [5 marks] Under what condition will a Binary Search Tree have bad performance? Explain why.

Condition: When the BST is unbalanced, so that the maximum depth of the tree is O(n).

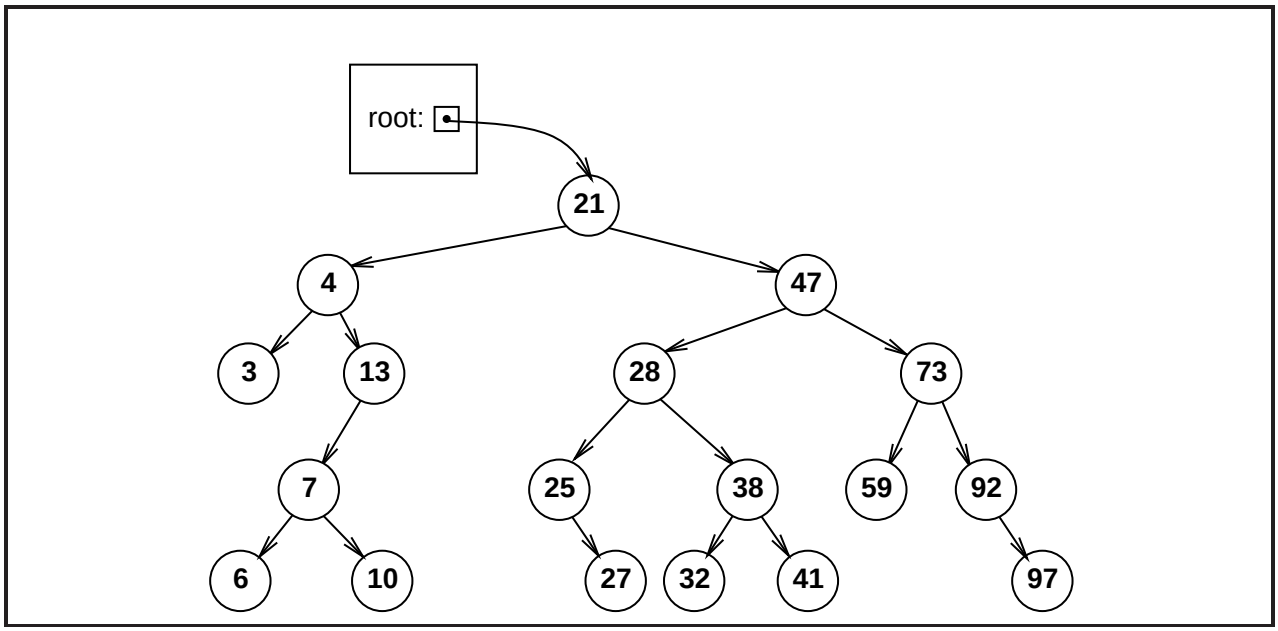
Explanation: When adding a node that belongs near the end of the long branch, the cost will be O(n). This bad performance can happen if the items added are already in sorted order.

(Question 5 continued on next page)

Student ID: .....

(Question 5 continued)

(c) [8 marks] In the blank space below, draw what the following binary search tree would look like if the values 3, 13, 47, and 21 were removed, in that order. Use the removal algorithm presented in class.



(Question 5 continued on next page)

Student ID: .....

(Question 5 continued)

(d) [18 marks] Given the declarations below for the BSTSet and BSTNode classes, complete the add method below (for the BSTSet class) and the add method on the facing page (for the BSTNode class) for adding a new value to a Set implemented using a Binary Search Tree.
Assume that the value is not already in the set. You may compare values using the lessThan, greaterThan, and equals methods.

```
public class BSTSet<E> {
    // data fields
    private BSTNode<E> root;
        ⋮
    // BSTNode class
    private class BSTNode<E> {
        private E value;
        private BSTNode<E> left;
        private BSTNode<E> right;

        public BSTNode(E v){
            value = v;
        }

        ⋮

    }
}
```

```
// in the BSTSet class ...
    /* Adds an item to the set. Assumes that item is not present */
    public void add (E item) {
        if (item == null) return;
        if (root == null){
            root = new BSTNode<E> (item);

        } else {
            root.add(item)
        }
    }
```

(Question 5 continued on next page)

Student ID: .....

(Question 5 continued)

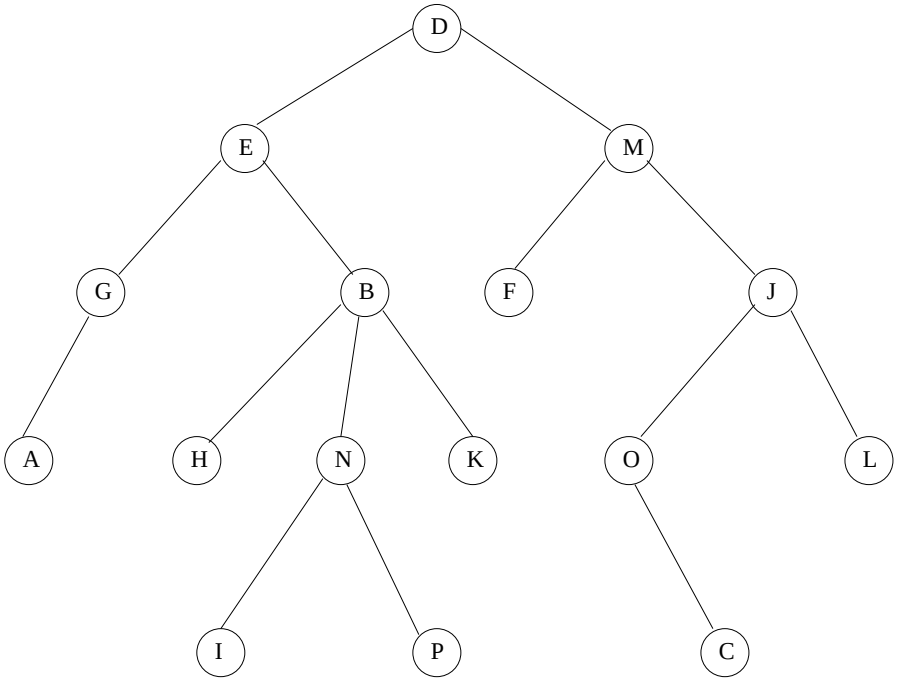
```
// in the BSTNode class ...
   /* Adds an item to the binary search tree. */
   /* Assumes that item is not present */
   public boolean add(E item){
      if (item.lessThan(value)){
         if (left == null)
            left = new BSTNode<E>(item);
         else
            left.add(item);
      }
      else {
         if (right == null)
            right = new BSTNode<E>(val);
         else
            right.add(val);
      }









   }
```

Student ID: .....

Question 6. Tree Traversals

[20 marks]

(a) [5 marks] Write out the order in which nodes would be visited by a breadth-first, left-to-right traversal of the following general tree.



(b) [3 marks] What is the worst-case Big-O (asymptotic) cost of finding a particular value in a binary tree (not a binary search tree!) of maximum depth d?

Cost: $O(2^d)$

(Question 6 continued on next page)

Student ID: .....

(Question 6 continued)

(c) [12 marks] Complete the following numLeaves() method so that it returns the number of leaf nodes in a GeneralTree.

```
public class GeneralTreeNode<E> {
    private Set<GeneralTreeNode<E>> children;
    private E value;

        ⋮

    /*Returns true if the node is a leaf node*/
    public boolean isLeaf(){
        return (children==null || (children.isEmpty()));
    }
    public int numLeaves(){
        int ans = 0;
        if (isLeaf())
            ans = 1;
        else
            for (GeneralTree<E> ch : children)
                ans = ans + ch.numLeaves();
        return ans;









    }
}
```

Student ID: .....

Question 7. Heaps [16 marks]

(a) [3 marks] Define a partially ordered tree.

(b) [3 marks] Define a complete binary tree.

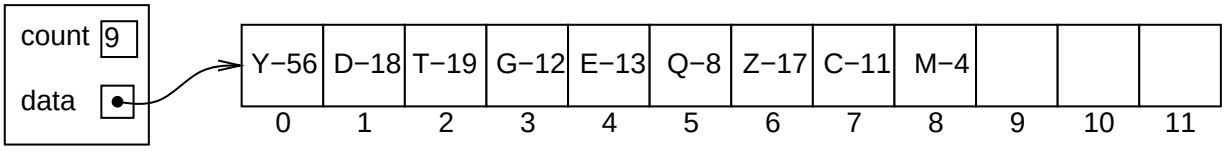
(Question 7 continued on next page)

Student ID: .....

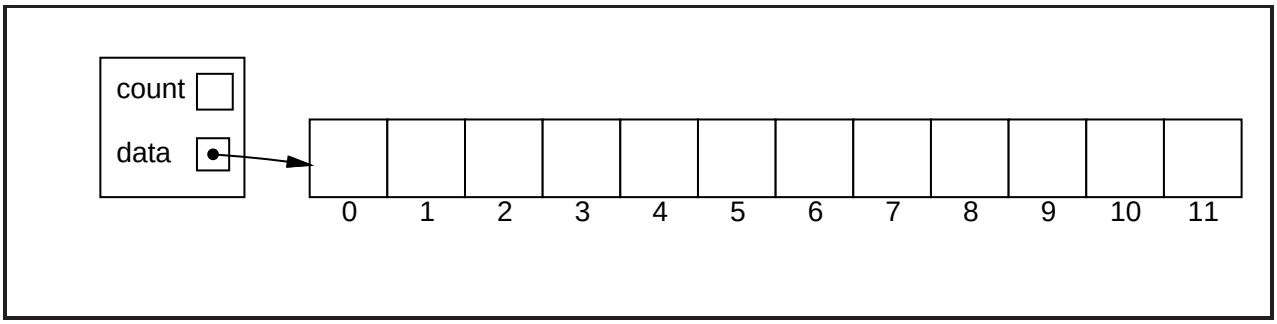
(Question 7 continued)

The HeapQueue class is an implementation of priority queues using a complete, partially ordered binary tree in an array.

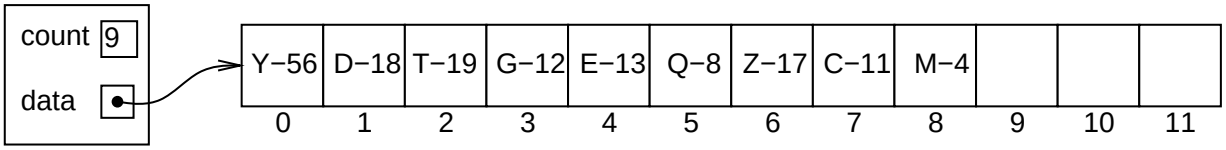
Consider the following HeapQueue of letter-integer pairs, where the integer represents the priority (larger numbers are higher priority).



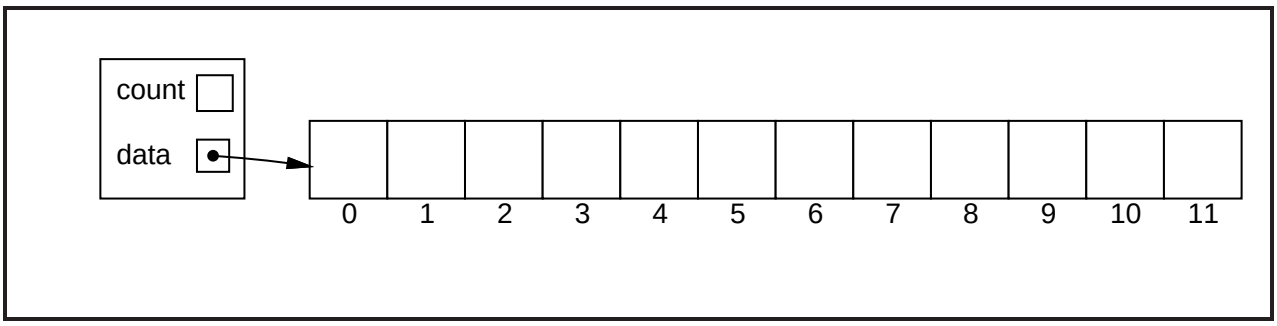
(c) [5 marks] Show the state of the HeapQueue if poll() is called on it.
Hint: you may find it easier to first draw the heap as a tree.



(d) [5 marks] Consider the same HeapQueue of letter-integer pairs



Show the state of the HeapQueue if the pair Q-23 is added to the HeapQueue:



Student ID: .....

Question 8. Hashing [18 marks]

(a) [2 marks] If we are adding a value to a Hash Table using buckets, and the first bucket we look at already contains a value, where do we put the value?

In the bucket

(b) [6 marks] Under what conditions will a Hash Table using linear probing have bad performance? Explain why.

Condition: Whenever the hashtable is close to full.

Explanation: Because there will be many collisions, so that adding an item will require looking at many cells.
(Extra marks: Also, if the hash function is bad so that many items hash to the same value.)

(c) [5 marks] What is wrong with the following hash function?

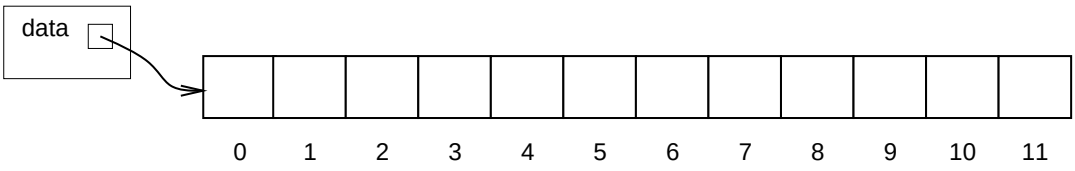
```
public int hashFunction(String key) {
    int hash = 1;
    char[ ] characters = key.toCharArray();
    for (int i = 0; i < characters.length; i++)
        hash = hash + (int) (characters[i] * 256 * Math.random());
    return (hash % data.length); // data is the hash table array
}
```

Student ID: .....

The use of Math.random means that each time a value is hashed, it will produce a different value. This is useless because it means that the hash table will not be able to find an item again after it has inserted it

(d) [5 marks] Draw the contents of the array after the following 8 values are inserted (in the order shown) into a Hash Table using quadratic probing (where the probing sequence is hash, hash+1^2, hash+2^2, hash+3^2, ...).

value	*hashed index:*
'Q'	$\Longrightarrow$ 7
'A'	$\Longrightarrow$ 0
'J'	$\Longrightarrow$ 3
'K'	$\Longrightarrow$ 3
'E'	$\Longrightarrow$ 0
'G'	$\Longrightarrow$ 1
'L'	$\Longrightarrow$ 10
'B'	$\Longrightarrow$ 9



Student ID: .....

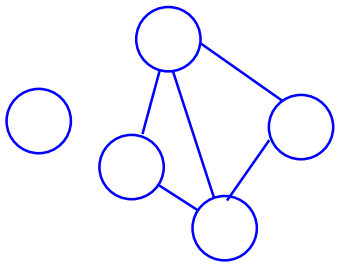
Question 9. Graphs

[21 marks]

(a) [2 marks] What is the difference between a directed graph and an undirected graph?

In a directed graph, the edges have an arrow,
representing a connection in only one direction between the two nodes.
In an undirected graph, the edges have no arrow, and the edge
represents a bidirection connection between the nodes

(b) [4 marks] Draw a disconnected graph containing five nodes and five edges.

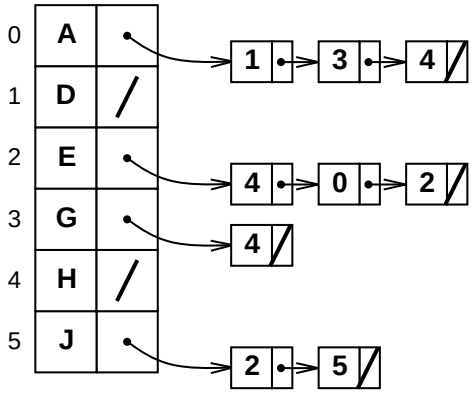


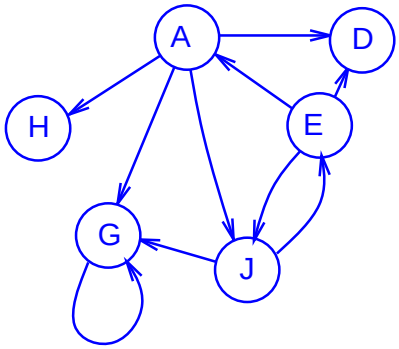
(Question 9 continued on next page)

Student ID: .....

(Question 9 continued)

(c) [5 marks] Draw the directed graph that is represented by the following adjacency list representation.





(Question 9 continued on next page)

Student ID: .....

(Question 9 continued)

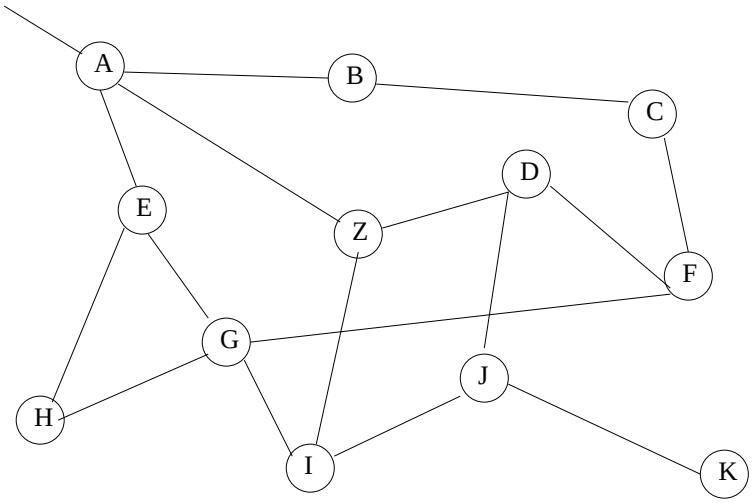
Suppose that the nodes of a graph specified using a linked structure (not an adjacency list or matrix!) are represented using a GraphNode class with the following methods:

```
public String getValue();                    //returns the value in a node
public Set<GraphNode> getNeighbours();       //returns the neighbours of a node
```

Consider the following (incorrect) printAll method for printing out the values in a graph using a breadth-first traversal.

```
private void printAll(GraphNode start) {
    Queue<GraphNode> toVisit = new LinkedList<GraphNode> ();
    toVisit.offer(start);
    while (!toVisit.isEmpty()) {
        GraphNode node = toVisit.poll();
        System.out.println(node.getValue());
        Set<GraphNode> neighbours = node.getNeighbours();
        for (GraphNode next: neighbours)
            toVisit.offer(next);
    }
}
```

(d) [5 marks] What are the first 10 values that the printAll method will print out given the following graph if it is called on the node containing 'A'?
Assume that the neighbours of a node are given in alphabetical order.



(Question 9 continued on next page)

Student ID: .....

(Question 9 continued)

(e) [5 marks] Fix the printAll method below so that it does a correct breadth first traversal on graphs.

```
private void printAll(GraphNode start) {
    Queue < GraphNode > toVisit = new LinkedList < GraphNode > ();

    Set < GraphNode > visited = new HashSet < GraphNode > ();
    toVisit.offer(start);
    while (!toVisit.isEmpty()) {
        GraphNode node = toVisit.poll();


        visited.add(node);
        System.out.println(node.getValue());
        Set < GraphNode > neighbours = node.getNeighbours();
        for (GraphNode next: neighbours) {
            toVisit.offer(next);
        }

    }
}
```

****

Student ID: .....

SPARE PAGE FOR EXTRA ANSWERS

Cross out rough working that you do not want marked.
Specify the question number for work that you do want marked.

Appendices

Possibly useful formulas:

- $1 + 2 + 3 + 4 + \cdots + k \;=\; \frac{k(k+1)}{2}$
- $1 + 2 + 4 + 8 + \cdots + 2^k \;=\; 2^{k+1} - 1$
- $a + (a+b) + (a+2b) + \cdots + (a+kb) \;=\; \frac{(2a+kb)(k+1)}{2}$
- $a + as + as^2 + as^3 + \cdots + as^k \;=\; \frac{as^{k+1}-a}{s-1}$

Table of base 2 logarithms:

n	1	2	4	8	16	32	64	128	256	512	1024	1,048,576
$\log(n)$	0	1	2	3	4	5	6	7	8	9	10	20

Brief (and simplified) specifications of relevant interfaces and classes.

```
public class Scanner {
    public boolean hasNext();          //there is more to read
    public String next();              //return the next token (word)
    public String nextLine();          //return the next line
    public int nextInt();              //return the next integer
}
public interface Iterator<E> {
    public boolean hasNext();
    public E next();
    public void remove();
}
public interface Comparable <E>{
    public int compareTo(E o);         //-ve if before o, 0 if same, +ve if after o
}
public interface Comparator <E>{
    public int compare(E o1, E o2); //-ve if o1 before o2, 0 if same, +ve if o1 after o2
}
public class Math{
    public static double random();   //return a random number between 0.0 and 1.0
}
```

```
public interface Collection <E>{
    public boolean isEmpty ();
    public int size ();
    public Iterator<E> iterator ();
}
public interface List <E>extends Collection <E>{
    // Implementations: ArrayList
    public E get (int index);
    public void set (int index, E element);
    public void add (E element);      //Add to end of list
    public void add (int index, E element);
    public void remove (int index);
    public void remove (Object element);
}
public interface Set extends Collection <E> {
    // Implementations: ArraySet, SortedArraySet, HashSet
    public boolean contains (E element);
    public void add (E element);
    public void remove (Object element);
}

public interface Map <K, V> {
    // Implementations: HashMap, TreeMap, ArrayMap
    public V get (K key);             //returns null if no such key
    public void put (K key, V value);
    public void remove (K key);
    public Set<Map.Entry<K, V> > entrySet ();
}
public interface Queue <E>extends Collection <E>{
    // Implementations: ArrayQueue, LinkedList
    public E peek ();                 //returns null if queue is empty
    public E poll ();                 //returns null if queue is empty
    public boolean offer (E element);
}
public class Stack <E>implements Collection <E>{
    public E peek ();                 //returns null if stack is empty
    public E pop ();                  //returns null if stack is empty
    public E push (E element);
}
public class Arrays {
    public static <E> void sort(E[ ] ar, Comparator<E> comp);
}
public class Collections {
    public static <E> void sort(List<E> list, Comparator<E> comp);
}
```