Answer four questions. If you answer more than four questions, your answer with the lowest mark will be ignored.

1. A ‘Rose Tree’ is a tree with an integer internal label and a List of subtrees, each of which is a Rose Tree. We call a List of Rose Trees a Tree Node (you will be asked to implement this as a linked list). We say that a Rose Tree is a leaf if it has no subtrees (i.e., its list of subtrees is the empty list, denoted ‘null’ below). A specification of Rose Trees, with an operation to get the list of integers at its leaves, is given below. First, we give a specification of lists of integers.

\[\text{mod LIST is} \]
\[\text{protecting INT.} \]
\[\text{sort List.} \]
\[\text{op empty: } \rightarrow \text{List.} \]
\[\text{op add: List Int } \rightarrow \text{List.} \]
\[\text{endm} \]

This specification is implemented by Vector<Integer>, with the constructor implementing ‘empty’ and addElement(Integer) implementing ‘add’. Rose Trees, with an operation, ‘getLeaves’, to get the integers at the leaves of a tree, are specified below:

\[\text{mod ROSETREE is} \]
\[\text{protecting LIST.} \]
\[\text{sort RoseTree.} \]
\[\text{sort TreeNode.} \]
\[\text{op roseTree: Int TreeNode } \rightarrow \text{RoseTree.} \]
\[\text{op null: } \rightarrow \text{TreeNode.} \]
\[\text{op treeNode: RoseTree TreeNode } \rightarrow \text{TreeNode.} \]
\[\text{op getLeaves: RoseTree } \rightarrow \text{List.} \]
\[\text{op addLeaves: RoseTree List } \rightarrow \text{List.} \]
\[\text{op loop: TreeNode List } \rightarrow \text{List.} \]
\[\text{var R: RoseTree.} \]
\[\text{var I: Int.} \]
\[\text{var L: List.} \]
\[\text{var T: TreeNode.} \]
\[\text{eq getLeaves(R) = addLeaves(R, empty).} \]
\[\text{eq addLeaves(roseTree(I, null), L) = add(L, I).} \]
\[\text{cq addLeaves(roseTree(I, F), L) = loop(F, L) if F /= null.} \]
\[\text{eq loop(null, L) = L.} \]
\[\text{eq loop(treeNode(R, T), L) = loop(T, addLeaves(R, L)).} \]
\[\text{endm} \]

(a) Give Java implementations of classes RoseTree and TreeNode (note that both classes refer to each other, which is perfectly acceptable in Java). Marks will be awarded for:
• correctly implementing TreeNode as a linked list structure in Java (hint: all that is needed are the fields and a constructor implementing the operation ‘treeNode’) [5 marks]
• correctly implementing the operation ‘roseTree’ as a constructor for class RoseTree [3 marks]
• correctly implementing getLeaves and its ‘helper’ method, addLeaves (note that the Maude specification uses another helper operation, ‘loop’: this does not need to be implemented as a method; the most direct way to implement the specified functionality in Java is by means of a while-loop). [10 marks]

(b) Briefly say why your implementation of ‘addLeaves’ behaves in the way specified by the Maude equations [7 marks]

2. Sometimes it is useful to have operations that return a pair of values. For example, integer division by two can be thought of as an operation that returns a pair consisting of an integer (the ‘factor’) and a digit (0 or 1, the ‘remainder’). For example, given input 13, division by two gives factor 6 and remainder 1; given input 6, division by two gives factor 3 and remainder 0; given input 3, we get factor 1 and remainder 1; and given input 1, we get factor 0 and remainder 1. As this example illustrates, if we repeatedly apply division by two to the resulting factors, and gather the remainders into a list, we get the list of remainders: 1 1 0 1, which gives the binary representation of the number 13.

(a) Give an implementation in Java of a generic class Pair<A,B>, where A is the type of the first component in the pair, and B the type of the second component of the pair. [6 marks]

(b) Write a generic interface, PairFunction, specifying a method compute that takes input belonging to some type A and returns a pair whose first component is of type A, and whose second component belongs to some type B. [3 marks]

(c) Write a generic method iterate that takes a PairFunction<A,B>, f, and an integer, n, as parameters, and returns a PairFunction<A,Vector<B>>, which will repeatedly apply f, n times, gathering the ‘B results’ into a Vector (these ‘B results’ come from the second components of the pairs returned by f’s compute method; on each repeated application, f’s compute method is applied to the first component). [10 marks]

(d) Generic types are implemented in Java by a process called erasure, which removes generic type parameters. What would the results of this process be for:
   i. the interface PairFunction? [2 marks]
   ii. the generic method iterate? [4 marks]
3. An implementation of bounded stacks is given below; a stack can hold at most five integers. The class has a method remove(int) that pops the stack a given number of times.

```java
public class Stack {
    private int[] values = new int[5];
    private int top = 0;

    public boolean isEmpty() {
        return top <= 0;
    }

    public int pop() {
        return values[--top];
    }

    public void push(int i) {
        values[top++] = i;
    }

    public void remove(int i) {
        while (i > 0) {
            pop();
            i--;
        }
    }

    public static void main(String[] args) {
        Stack s = new Stack();
        s.push(3);
        s.remove(2);
    }
}
```

(a) Briefly describe the ‘method-call stack’ in the Java interpreter. [3 marks]

(b) What happens when the main method in the Stack class is executed? Describe the state of the method-call stack during the execution of this main method. [5 marks]

(c) Briefly describe the differences between ‘checked’ and ‘unchecked’ exceptions. [4 marks]

(d) Write a checked exception class, EmptyStackException, and modify the pop() method in class Stack so that it throws an EmptyStackException if the stack is empty. What other changes would be necessary to the class? [7 marks]

(e) With these changes, is the property $0 \leq \text{top}$ a class invariant? Justify your answer. [6 marks]
4. (a) In the context of multi-threaded programs, what is meant by ‘interference’? [2 marks]

(b) Consider the following program, which creates two threads sharing a Stack instance (using the class from Question 3).

```java
class ThreadTest implements Runnable
{
    private Stack stack;

    ThreadTest(Stack s)
    {
        stack = s;
    }

    public void run()
    {
        for (int i = 0; i < 200; i++)
        {
            if (stack.isEmpty())
            {
                stack.push(i);
            }
            else
            {
                System.out.println("top: " + stack.pop());
            }
        }
    }

    public static void main(String[] args)
    {
        Stack s = new Stack();
        Thread t1 = new Thread(new ThreadTest(s));
        Thread t2 = new Thread(new ThreadTest(s));
        t1.start();
        t2.start();
    }
}
```

(c) Describe how interference might arise in this example. [7 marks]

(d) Describe how synchronization is used in Java to prevent interference. [7 marks]

(e) Briefly describe the difference between a synchronized block of code and a synchronized method. [3 marks]

(f) How would you modify the example of part (a) in order to prevent interference? [6 marks]
5. (a) What is meant by the term ‘Abstract Data Type’? \[5 \text{ marks}\]

(b) The Abstract Data Type of **Sorted Lists** consists of lists of integers in increasing order (small values at the start of the list, large values at the end). The empty list, ‘nil’, is a Sorted List, and there is an operation, ‘insert’ that puts an integer into a Sorted List at the appropriate place (so that the list remains sorted). There is also an operation, ‘merge’, that takes two Sorted Lists and gives the Sorted Lists containing all the elements in the two given lists, in increasing order. There is also a private operation, ‘add’, that adds an integer to the start of a Sorted List.

Give a Maude specification of Sorted Lists, with equations that specify the behaviour of ‘insert’ and ‘merge’ \[15 \text{ marks}\]

(c) Describe how Maude will use the equations given in your answer to part (b) to reduce the term insert(2, insert(1, nil)). \[5 \text{ marks}\]