PROBLEM 1 : (Oversees, Refugees, Guarantees, Search-trees (30 points))

The tree below is a search tree.

Part A (4 points)
What is the pre-order traversal of the tree? (The first value printed is macaque)

Part B (2 points)
Provide a word (it doesn’t have to be a real word, it must contain at least four letters) that could be inserted as a left-child of monkey so that the tree is still a search tree.

Part C (2 points)
Show by drawing where a node with gibbon would be inserted into the tree.

Part D (4 points)
The tree above has a height of four and has three leaves. Draw a search tree with the same values, a height of four, and which has four leaves. Draw the tree on the previous page.

continued →
Part E (4 points)
The method `printQ` below prints one line for every node in a tree. The first value printed when called with the tree above is the string `macaque`. What is the complete output?

```java
public static void printQ(TreeNode root){
    Queue<TreeNode> q = new LinkedList<TreeNode>();
    if (root != null){
        q.add(root);
    }
    while (q.size() != 0){
        root = q.remove();
        System.out.println(root.info);
        if (root.left != null) q.add(root.left);
        if (root.right != null) q.add(root.right);
    }
}
```

Part F (4 points)
The method `printS` below prints one line for every node in a tree. The first value printed when called with the tree above is the string `macaque`. What is the complete output?

```java
public static void printS(TreeNode root){
    Stack<TreeNode> st = new Stack<TreeNode>();
    if (root != null){
        st.push(root);
    }
    while (st.size() != 0){
        root = st.pop();
        System.out.println(root.info);
        if (root.right != null) st.push(root.right);
        if (root.left != null) st.push(root.left);
    }
}
```

continued →