**Section 03: Binary Search Trees and AVL Trees**

1. **Binary Search Trees**

Let a binary search tree be defined by the following class:

```java
public class IntTree {
    private IntTreeNode overallRoot;

    // constructors and other methods omitted for clarity

    private class IntTreeNode {
        public int data;
        public IntTreeNode left;
        public IntTreeNode right;

        // constructors omitted for clarity
    }
}
```

(a) Given a binary search tree (as defined above), write a method (sufficient to present a pseudocode) to output the elements in sorted order.

(b) Let \( n \) be the number of nodes in a binary tree. What is the runtime of your method from Question 1(a) as a function of \( n \)?

(c) Draw the binary search tree after the execution of each operation in the sequence

```
insert(10), insert(20), insert(15), insert(2), insert(25), insert(22), insert(50),
remove(2), remove(20), remove(10), insert(33), remove(50), insert(20).
```

Assume that the tree is empty before the execution of the sequence.

2. **AVL Trees**

(a) Identify if the following trees are AVL trees. Explain your answer.

(i) Tree 1

![Tree 1](image)

Assume that the tree is empty before the execution of the sequence.
(ii) Tree 2

(iii) Tree 3

(b) Draw an AVL Tree as each of the following keys are added in the order given. Show intermediate steps.

(i)

\{13, 17, 14, 19, 22, 18, 11, 10, 21\}

(ii)

\{1, 2, 3, 4, 5, 6\}