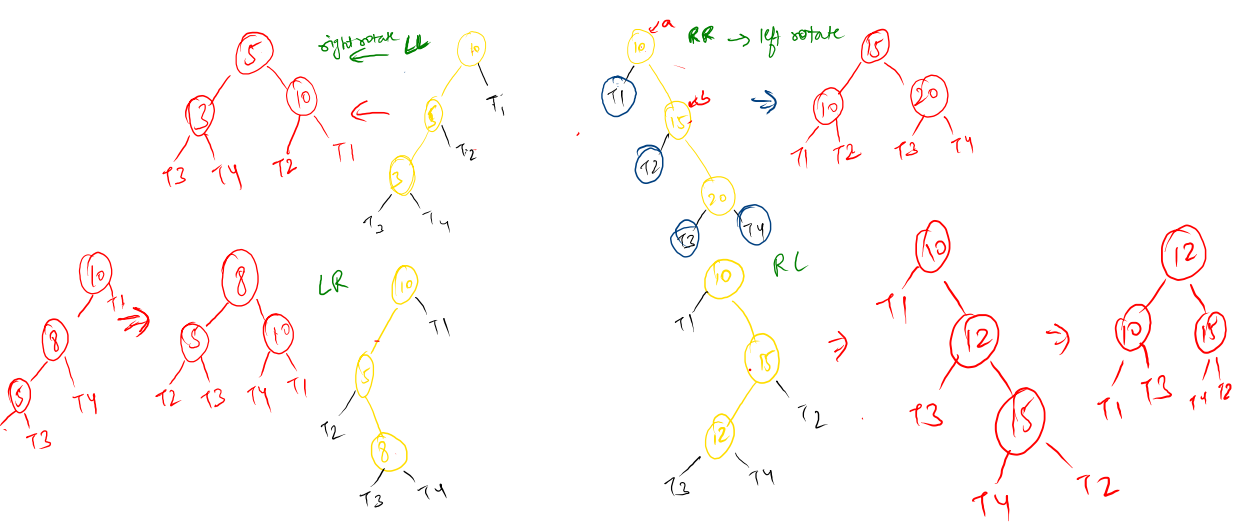


```
public Node leftrotate(Node a) {
    Node b = a.right;
    Node t2 = b.left;

    Node root = b;

    b.left = a;
    a.right = t2;

    a.height = Math.max(height(a.left), height(a.right)) + 1;
    b.height = Math.max(height(b.left), height(b.right)) + 1;

    return root;
}

public Node rightrotate(Node a) {
    Node b = a.left;
    Node t2 = b.right;

    Node root = b;
    b.right = a;
    a.left = t2;

    a.height = Math.max(height(a.left), height(a.right)) + 1;
    b.height = Math.max(height(b.left), height(b.right)) + 1;

    return root;
}
```

```
if (node.data > data) {
    node.left = insertToAVL(node.left, data);
} else if (node.data < data) {
    node.right = insertToAVL(node.right, data);
}

int leftheight = height(node.left);
int rightheight = height(node.right);

node.height = Math.max(leftheight, rightheight) + 1;

int diff = leftheight - rightheight;

Node newroot = node;
if (diff > 1 && node.left.data > data) { // LL
    newroot = rightrotate(node);
} else if (diff > 1 && node.left.data < data) { // LR
    node.left = leftrotate(node.left);
    newroot = rightrotate(node);
} else if (diff < -1 && node.right.data < data) { // RR
    newroot = leftrotate(node);
} else if (diff < -1 && node.right.data > data) {// RL
    node.right = rightrotate(node.right);
    newroot = leftrotate(node);
}

return newroot;
```

O(logn)

1 → 0001
2 → 0010
3 → 0011
4 → 0100
5 → 0101
6 → 0110
7 → 0111
8 → 1000
9 → 1001
10 → 1010
11 → 1011
12 → 1100