Ambere Mishra
1BM18CS013
ADS

• AVL tree insertion -

```
Node insert (root, key)
{
    if (root == null)
        return (new Node(key));
    if (key < root.val)
        root.left = insert(root.left, key)
    else if (key > root.val)
        root.right = insert(root.right, key)
    else
        return root

    root.height = 1 + Math.max(height(root.left), height(root.right
    int balfactor = get(root);
    if (balfactor > 1 && key < root.left.val)
            return rightRotate(root)

    if (balfactor > 1 && key > root.left.val)
        {   root.left = leftRotate(root)
            return rightRotate(root);
        }
```

Ambere

```
if (bodfactor < -1 && key > root.right.val)
        return leftRotate(root);

if (bodfactor < -1 && key < root.right.val)
{
        root.right = rightRotate(root.right)
        return leftRotate(root);
}

        return root;
}
Node delete(root, key) - Deletion
{
    if (root == null)
        return root;

    if (key < root.val)
        root.left = delete(root.left, key);

    else if (key > root.val)
        root.right = delete(root.right, key);

    else
    {
        if ((root.left == null) || (root.right == null))
        {
            Node t = null;
            if (t == root.left)
                t = root.right
            else
```

Teacher's Signature :

Ankur Mohan
IBM & CSE 17
ADS Lab

```
t = root.left;
if (t==null)
 { t = root;
   root = null;
 }
else
  root = t;
}
else
{
  Node t = successor(root.right) //In-order successor
   root.val = t.val;
   root.right = delete(root.right, t.val)
  }
if (root == null)
   return root
else
{
   Perform balancing same as in
   insertion
}
return root;
```

Ankur