Ajay Mittur
IBM18CS002

AVL Tree

```
Node* insert(Node* node, int key) {
    if (!node) return new Node(k);
    if (key < node->key)
        node->left = insert(node->left, key);
    if (key > node->key)
        node->right = insert(node->right, key);
    node->height = 1 + max(node->left->height,
                           node->right->height);
    if (getbalance(node) > 1) {
        if (key < node->left->key)
            return rightRotate(node);
        else {
            leftRotate(node->left);
            return RightRotate(node);
        }
    }
    else if (getbance(node) < -1) {
        if (key > node->right->key)
            return leftRotate(node);
        else
            rightRotate(node->right);
            return leftRotate(node);
    }
    return node;
}
```

Ajay Mittur
1BM18IS006

Expt. No. ..... AVL Tree Page No. ..... Date .....

```
Node* delete(Node *root, int key) {
    if (!root) return root;
    if (key < root->key)
        root->left = delete(root->left, key)
    else if (key > root->key)
        root->right = delete(root->right, key)
    else {
        if (!root->left || !root->right) {
            Node * temp = root->left ? root->left : root->right;
            if (!temp) {
                temp = root; delete(root);
            else {                return root; //NULL
                *root = *temp;
                delete temp.
            }
            else {
            Node* temp = inorderSuccessor(root->right)
            root->key = temp->key;
            root->right = delete(root->right, temp
                                              ->key);
        }
    root->height = 1 + max(root->left->height,
                           root->right->height);
    int bf = getbalance(root);
    if (bf > 1) {
        if (getbalance(root->left) >= 0)
            return rightRotate(root);
        else
            leftRotate(root->left);
            return rightRotate(root)
    }
```

Teacher's Signature : _____

Ajay Nittur
(BM18CS006

```
if (bf < -1) {
    if (getbalance(root->right) <= 0)
        leftRotate (root)
    else {
        RightRotate (root->right)
        leftrotate (root)
    }
}
return root;
}
```