14/10/20

# AVL Tree

P. SAI DEEKSHITH
IBM18CS148

```
insert (Node * node, int data)

if node == NULL
    return newNode(data)
else if data < node -> data
    node -> lchild = insert(node -> lchild,
                            data);
else if data > node -> data
    node -> rchild = insert(
                node -> rchild,
                data);

node -> height = 1 + max(height(node ->
lchild), height(node -> rchild));

balance = getBalance(node)

if (balance < -1 && data > node ->
                    rchild -> data)
node = rotate(node) # left rotate
```

```
if (balance > 1 && data < node->lchild->data)
    Node = rightrotate(node)

if (balance > 1 && data > node->lchild->
                          data)

    Node->lchild = leftRotate(node->lch

    node = leftRotate(node)

    return node

Delete (Node* root, int data)
    if (root == NULL)
        return root
    if ( data < root->data)
    root->lchild = Delete(root->lchild
                          data)
    else if (data > root->data)
        root->rchild = Delete(root->rchild
                              data)

    else {
        if root->lchild is NULL &&
            root->rchild is NULL
        {
            temp = root->lchild? root->lchild
                                 root->rchild
```

if temp is NULL
temp = root
root = NULL

else root = temp
free(temp);}

else {
temp = minValueNode(root->rchild);
root->data = temp->data;
root->rchild = Delete(root->rchild, temp->data);

}

if root is NULL
return root

root->height = 1 + max(height(root->lchild), height(root->rchild));

balance = getBalance(root);

if (balance > 1 && getBalance(root->lchild) < 0)

root->lchild = leftRotate(root->lchild)

root = rightRotate(root)

if (balance > 1 and getBalance(root->lchild) >= 0)

root = rightRotate(root)

return root.