AKSHAY MITTUR
1BM18CS010

14/10/20

ADS Lab
Week 4

AVL Trees

Insertion:

```
function insert(Node* node, int key){
    if(!node) return newNode(key);
    if (key < node->key) node->left = insert(node->left, key);
    else if (key > node->key) node->right = insert(node->right, key);
    else return node;
    node->height = 1 + max(height(node->left), height(node->right));
    int balance = getBalance(node);
    if(balance > 1 && key < node->left->key) return rightRotate(node);
    if(balance < -1 && key > node->right->key) return leftRotate(node);
    if(balance > 1 && key > node->left->key){
        node->left = leftRotate(node->left);
        return rightRotate(node);
    }
    if(balance < -1 && key < node->right->key){
        node->right = rightRotate(node->right);
        return leftRotate(node);
    }
    return node;
}
```

AKSHAY M P TTJUR

14/10/20 1BM18CS010

Deletion:

```
[redacted] * deleteNode(Node* root, int key){
function↑
        if(!root) return root;
        if(key < root->key) root->left = deleteNode(root->left, key);
    else if(key > root->key) root->right = deleteNode(root->right, key);

    else{
        if(!root->left || !root->right){
                Node * temp = root->left? root->left : root->right;
                if (temp == NULL){
                        temp = root;
                        root = NULL;
                }
                else *root = *temp
                free(temp)
        }
        else{
                Node* temp = minNode(root->right);
                root->key = temp->key;
                root->right = deleteNode(root->right, temp->key);
        }
    }
    if(!root) return root;
    root->height = 1 + max(height(root->left), height(root->right));
    int balance = getBalance(root);
    if (balance > 1 && getBalance(root->left)>=0)
                return rightRotate(root);
    if (balance >1 && getBalance(root->left)<0){
                root->left = leftRotate(root->left);
                return rightRotate(root);
    }
```

AKSHAY MITTUR
1BM18CS010

14/10/20

```
if(balance <-1 && getBalance(root->right) <=0)
        return leftRotate(root);
if (balance <-1 && getBalance(root->right) >0) {
        root->right = rightRotate(root->right);
        return leftRotate(root);
}
return root;
}
```