Lab4 14/10/2020 1BM18CS153

vinay kumar

DATE: PAGE:

Insertion & deletion operation on AVL tree

Insert

```
Node insert (Node node, int key) {

  if (node == null)
      return (new Node (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 = get_Balance (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);
    return rightRotate (node);
}
```

DATE PAGE

```
if (balance < -1 || key < node.right.key) {
    node.right = rightRotate(node.right);
    return leftRotate(node);
}
return node;
}
```

Delete

```
Node deleteNode(Node root, int key) {
if (root == null)
    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 == null) || root.right == null)
    {
        Node temp = null;
        if (temp == root.left)
            temp = root.right;

        else
            temp = root.left;
```

DATE: PAGE:

```
if (temP == null)
{
    temp = root;
    root = null;
}
else
    root = temp;
}
else {
    Node temp = minValueNode (root.right);
    root.key = temp.key;
    root.right = deleteNode (root.right, temp.key)
}
}
if (root = null)
    return root;
root.height = max (height (root.left),
                   height (root.right)) + 1;

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);
}
```

DATE: PAGE:

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