Name: Anusree Manoj K
USN: 1BM18CS017
Date: 14-10-2020

LAB 5 Program 4 : AVL Trees insertion and deletion

```
Node* insert (Node* node, int key)
{
    if (node == NULL)
        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;
}
```

Anusree

Anusree Manoj
IBM18CS017

```
Node* delete (Node* root, int key)
{
    if (root == NULL)
        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 == NULL || root->right == NULL)
        {
            Node *temp = root->left ? root->left :
                                      root->right
            if (temp == NULL)
            {
                temp = root;
                root = NULL;
            }
            else
                *root = *temp;
            free (temp);
        }
        else
        {
            Node * temp = minValueNode (root->right);
            root->key = temp->key;
            root->right = delete (root->right, temp->key);
        }
    }
```

Anusree Manoj K
IBM18CS017

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

Anusree

Anusree Manoj K
1BM18CS017

```
Node * rightRotate (Node *node)
{
    Node *x = node->left;
    Node *t = ~~node->right~~ x->right;

    x->right = node;
    node->left = t;

    node->height = max (height(node->left),
                        height(node->right)) + 1;
    x->height = max (height(x->left), height(x->right))
                + 1;

    return x;
}

Node *leftRotate (Node *node)
{
    Node *x = node->right;
    Node *t = x->left;

    x->left = node;
    node->right = t;

    node->height = max (height(node->left),
                        height(node->right)) + 1;

    x->height = max (height(x->left),
                     height(x->right)) + 1;

    return x;
}
```

Anusree