Nidhish Lakhmana
IBM18CS079

Red Black Tree insertion

```
rotateLeft (&root, pt) {
    Node *pt_r = pt->right;
    pt->right = pt->right->left;

    if (pt->right != NULL)
        pt->right->parent = pt;

    pt_r->parent = pt->parent;

    if (pt->parent == NULL)
        root = pt_r;

    else if (pt == pt->parent->left)
        pt->parent->left = pt_r;

    else
        pt->parent->right = pt_r;

    pt_r->left = pt;
    pt->parent = pt_r;
}

rotateright (&root, &pt)
{
    Node *pt_left = pt->left;
```

```
pt->left = pt->l->right;

if (pt->left != NULL)
    pt->left->parent = pt;

pt->l->parent = pt->parent

if (pt->parent == NULL)
    root = pt->left;

else if (pt == pt->parent->left)
    pt->parent->left = pt->l;

else
    pt->parent->right = pt->l;

pt->l->right = pt;
pt->parent = pt->l;
}
```

```
insert (to int & data) {
    Node *pt = new Node (data);

    root = BSInsrt (root, pt);      root = BSInsrt (root, pt);
    ~~bst~~ Insert fix (root, pt);      Insrt fix (root, pt);

    ~~if (root == NULL)~~

        {


BSInsrt (root, pt) {
    if (root == NULL)
        return pt;

    if (pt -> data < root -> data)
    {
        root -> left = BSTInsrt (root -> left, pt);
        root -> left -> parent = root;
    }
    else if (pt -> data > root -> data)
    {
        root -> right = BSTInsrt (root -> right, pt)
        root -> right -> parent = root;
    }

    return root;
}
```