Red-Black Tree
Insertion

GURU NANMA
1BM18CS031

```
fun inorderHelper (root)
        if root == NULL
            return
        inorderHelper(root -> left)
        print root -> data
        inorderHelper(root -> right)

fun BSTInsert (root, pt)
        if root == NULL
            return pt

        if pt -> data < root -> data
            root -> left = BSTInsert(root -> left, pt)
            root -> left -> parent = root
        else if (pt -> data > root -> data)
            root -> right = BSTInsert(root -> right, pt)
            root -> right -> parent = root

        return root

fun levelOrderHelper (root)
        if root == NULL
            return
        queue<Node * > q
        q.push(root)
```

GURU NANMA
1BM18CS031

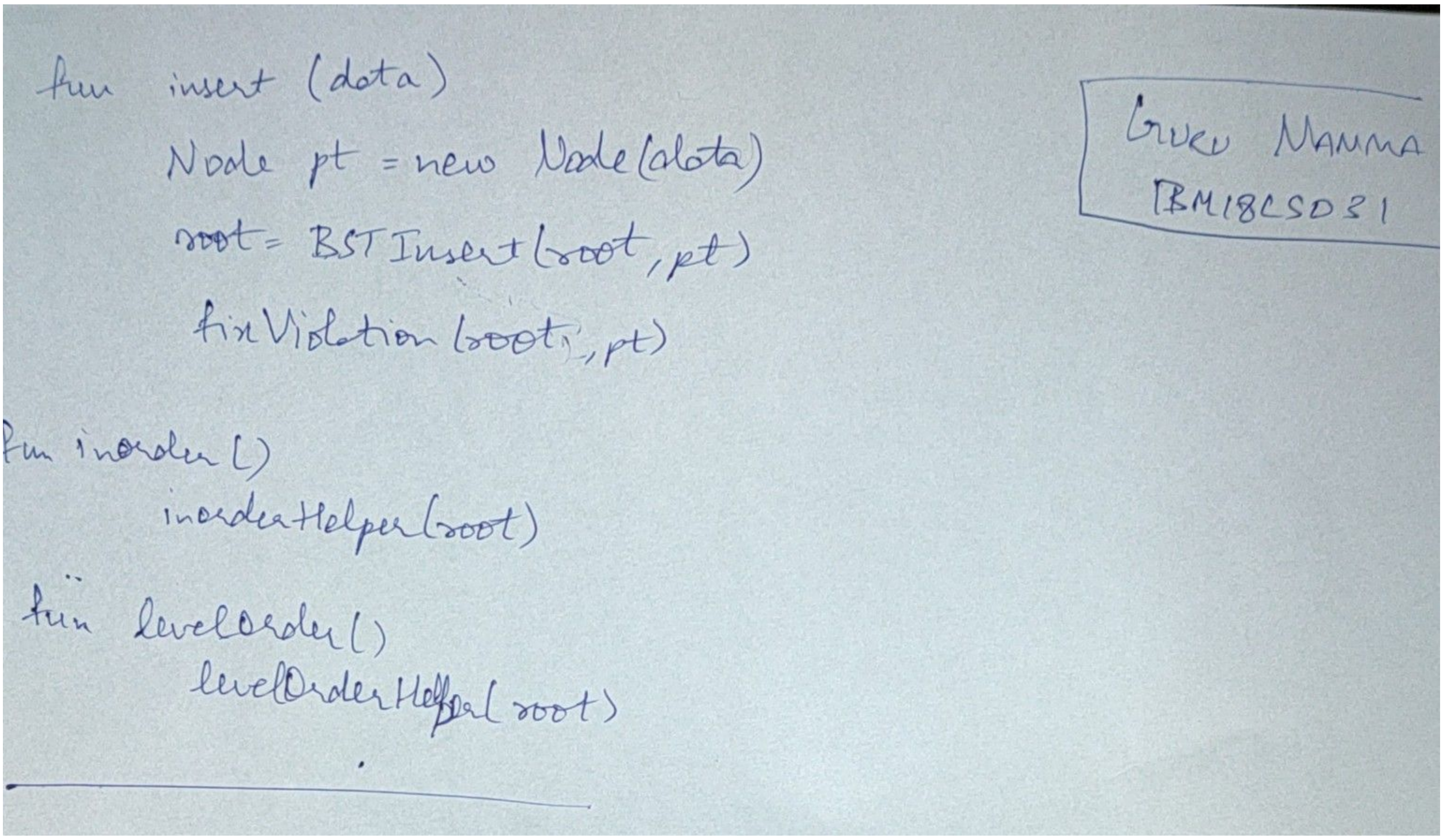
```
while (!q.empty())
      Node *temp = q.front()
      print temp->data
      q.pop()

      if (temp->left != NULL)
            q.push (temp->left)
      if (temp->right != NULL)
          q.push (temp->right)

fun rotateLeft( root , pt)
        pt-right = pt->right
        pt->right = pt-right->left

        if (pt->right != NULL)
              pt->right->parent = pt
        pt-right->parent = pt->parent
        if (pt->parent == NULL)
              root = pt-right
        else if (pt == pt->parent->left)
              pt->parent->left = pt-right
        else
            pt->parent->right = pt-right

        pt-right->left = pt
        pt->parent = pt-right

fun rotateRight (root , pt)
    Node pt_left = pt->left
     pt->left = pt_left->right
     if (pt->left != NULL)
               pt->left->parent = pt

     pt_left->parent = pt->parent
```

GURU NANN
IBM18CS031

```
    if (pt → parent == NULL)
        root = pt_left
    else if (pt == pt → parent → left)
        pt → parent → left = pt_left
    else
        pt → parent → right = pt_left .

    pt_left → right = pt
    pt → parent = pt_left

fun fixViolation (root, pt)
    Node parent-pt = NULL
    Node grand-parent-pt = NULL
    while ((pt != root) and (pt → color != BLACK)
                and (pt → parent → color == RED))
            parent-pt = pt → parent
            grand-parent-pt = pt → parent → parent
            if (parent-pt == grand-parent-pt → left)
                Node uncle-pt = grand-parent-pt → right
                if (uncle-pt != NULL and uncle-pt
                                            → color == RED
                    grand-parent-pt → color = RED
                    parent-pt → color = BLACK
                    uncle-pt → color = BLACK
                    pt = grand-parent-pt
```

GURU NANMA
1BM18CS031

```
        else
          if (pt == parent_pt -> right)
              rotateLeft (root, parent_pt)
              pt = parent_pt
               parent_pt = pt -> parent

          rotateRight (root, grand_parent_pt)
          swap (parent_pt -> color,
                     grand_parent_pt -> color)
          pt = parent_pt


else
    Node uncle_pt = grand_parent_pt -> left

    if ((uncle_pt != NULL) and
                     (uncle_pt -> color == RED))

        grand_parent_pt -> color = RED
        parent_pt -> color = BLACK
        uncle_pt -> color = BLACK
        pt = grand_parent_pt
    else
      if (pt == parent_pt -> left)
          rotateRight (root, parent_pt)
          pt = parent_pt
          parent_pt = pt -> parent

      rotateLeft (root, grand_parent_pt)
      swap (parent_pt -> color,
                grand_parent_pt -> color)
      pt = parent_pt

root -> color = BLACK
```



```
fun  insert (data)
       Node pt = new Node(data)
       root = BSTInsert(root, pt)
        fixViolation(root, pt)

fun inorder ()
       inorderHelper(root)

fun levelOrder()
        levelOrderHelper(root)
```

GURU MANNA
1BM18CS031