```
void RBTree::Insert (int data)
{
    Node *pt = new Node(data);

    root = BSTInsert(root, pt);

    fixViolation(root, pt);
}

void fixViolation (Node *&root, Node *&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;
        grandparent_pt = pt->parent->parent;

        if (parent_pt == grand_parent_pt->left)
        {
            Node *uncle_pt = grand_parent_pt->right;

            if (uncle_pt != NULL && uncle_pt->colour == RED)
            {
                grandparent_pt->color = RED;
                uncle_pt->colour = parent_pt->color = BLACK;
```

Date .....

Expt. No. .....

Page No. .....

```
        pt = grand_parent_pt;
    }

    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 -> colour);
        pt = parent_pt;
    }
}

else
{
    Node * uncle_pt = grand_parent_pt -> left;

    if ((uncle_pt != NULL) and (uncle_pt -> colour == RED))
    {
        grand_parent_pt -> color = RED;
        parent_pt -> color = BLACK;
        uncle_pt -> colour = BLACK;
        pt = grand_parent_pt;
    }
```

Teacher's Signature : _____

```
        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;
}
```