MANU.N.Y
IBM18CS053

```
void RBTree :: insert (const int &data){
    Node *ptr = new node (data);
    root = BST Insert (root, ptr);
    fix violation (root, ptr);
}

Node * BST Insert (Node *root, Node *ptr){
    if(root==NULL)
        return ptr;
    if (ptr -> data < root -> data){
        root -> left = BST Insert (root -> left, ptr)
        root -> left -> parent = root;
    }
    else if (ptr -> data > root -> data){
        root -> right = BST Insert (root -> right, ptr)
        root -> right -> parent = root;
    }
    return root;
}
```

Case A: Parent of ptr is left child of grandparent of ptr

Case 1: Uncle node of ptr is also red
- colour change

Case 2: ptr is right child of its parent left
- rotation required

Case 3: ptr is left child of its parent right
- rotation is required

Case B: parent of ptr is right child of grand parent of ptr

Case 1: Uncle of ptr is also red
- colour change

Case 2: ptr is right child of its parents left
- rotation required.