Nikhil.A.S
1BM18CS061
18/~~x~~11/2020.

```
void RBTree :: insert(const int &data){
    Node *ptr = new node (data);
    root = BSTInsert (root, ptr);
    fixviolation(root, ptr);
}

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

case A : Parent of ptr is left child of
  Grand parent of ptr

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

case 2:- ptr is right child of its
  parents 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
- color change
case 2: ptr is left child of its parents
right.
- rotation required.
case 3: ptr is right child of its parents
left
- rotation required.