Dhanraj. K
IBM18CS027

Insertion

```
Void TwoThreeTree :: insert(int k)
{
    if(root == NULL) {
        root = new TwoThreeNode(True);
        root -> Keys[0] = k;
        root -> n = 1;
    }
    else {
        if ( root->n == 2*t -1 ) {
            TwoThreeTree *newnode = newTwoThreeNode(false);
            newnode ->c[0] = root;
            newnode->SplitChild(0, root);
            int i =0 ;
            ~~if(k>k[0])~~
                i++
            if (newnode->Keys[0] < k)
                i++;
            newnode->C[i]->insertNonFull(k);
            root= newnode;
        }
        else {
            ~~newnode~~
            root->insertNonFull(k);
        }
    }
}

Void TwoThreeTree :: insertNonFull(int k) {
    int i = n-1;
    if (leaf == true) {
        while (i>=0 && Keys[i]>k)
        {   Keys[i+1] = Keys[i];
            i--; }
```

```
            keys[i+1] = k;
            n = n+1;
        } else {
            while (i >= 0 && keys[i] > k)
                i--;
            if (c[i+1] -> n == 2*t - 1)
            {
                splitChild (i+1, c[i+1]);
                if (keys[i+1] < k)
                    i++;
            }
            c[i+1] -> insertNonFull (k);
        }
    }
```

Deletion

```
void TwoThreeTree :: remove (int k)
{
    if (!root) {
        cout << "The tree is empty \n";
        return;
    }
    root -> remove (k);
    if (root -> n == 0) {
        TwoThreeNode *temp = root;
        if (root -> leaf)
            root = NULL;
        else
            root = root -> c[0];
        delete temp;
    }
    return;
}
```