11/11/20

Divyank Gupta
1BM18CS030

Write - Up

* B - Tree insertion

⊙ Pseudo-code :

```
void BTree :: insert (int K)
{
    if (root == NULL)
    then do
        root = new BTreeNode (t, true);
        root->keys[0] = K;
        root->n = 1

    else
      do
        if (root->n == 2*t-1)
          then do
            BTreeNode * s = new BTreeNode (t, false)
            s->C[0] = root;
            s->splitchild (0, root);
            int i = 0;
            if (s->keys[0] < K)
            then do
                  i = i+1
            s->C[i]->insertNonFull (K);
            root = s;

        else
          do
            root->insertNonFull (K);
}
```

Divyank

11/11/20

Divyanka Gupta
1BM18CS030

Write - Up

Continued . . .

```
void BTreeNode : insertNonFull (int K)
{
    int i = n-1;
    if (leaf == true)
    then do
        while (i>=0 && keys[i]> K)
        do
            keys[i+1] = keys[i];
            i = i-1;

        keys[i+1] = K;
        n= n+1;

    else
    do
        while (i>=0 && keys[i] > K)
        do
            i = i-1

        if (C[i+1]->n == 2* t-1)
        then do
            splitchild (i+1, C[i+1]);

            if (keys[i+1] < K)
            then do
                i= i+1

        C[i+1] -> InsertNonFull (K);

}
```

11/11/20

Divyank Gupta
1BM18CS030

Write - Up

Continued....

```
void BTreeNode::splitChild(int i, BTreeNode *y)
{
    BTreeNode *z = new BTreeNode(y->t, y->leaf);
    z->n = t-1

    for (int j=0; j<t-1; j++)
     do
        z->keys[j] = y->keys[j+t];

    if (y->leaf == false)
    then do
        for (int j=0; j<t; j++)
          z->C[j] = y->C[j+t];

    y->n = t-1

    for (int j=n; j>=i+1; j--)
     do
        C[j+1] = C[j];

    C[i+1] = z;

    for (int j=n-1; j>=i; j--)
     do
        keys[j+1] = keys[j];

    keys[i] = y->keys[t-1];

    n = n+1;
}
```


Divyank