Anusree Manoj K
IBM18CS017

Lab 7 - Program 7 - B-tree (insertion)

```
void BTree :: insert (int k)
{
    if (root == NULL)
    {
        root = new BTreeNode (t, true); //t is min degree
        root -> keys[0] = k;            // true => is leaf node
        root -> n = 1;
    }
    else {
        if (root -> n == 2*t - 1)
        {
            BTreeNode *s = new BTreeNode (t, false);
            s -> child[0] = root;
            s -> splitChild (0, root);
            int i=0;
            if (s -> keys[0] < k)
                i++;
            s -> child[i] -> insertNonFull (k);
            root = s;
        }
        else
            root -> insertNonFull (k);
    }
}
void BTree :: 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 (child[i+1]->n == 2*t-1)
        {
            splitChild (i+1, child[i+1]);
            if (keys[i+1] < k)
                i++;
        }
        child[i+1]->insertNonFull(k);
    }
}
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++)
        z->keys[j] = y->keys[j+t];
    if (y->leaf == false)
    {
        for(int j=0; j<t; j++)
            z->child[j] = y->child[j+t];
    }
    y->n = t-1;
    for (int j = n; j >= i+1; j--)
        child[j+1] = child[j];
    child[i+1] = z;
```

```
    for(int j=n-1; j>=i ;j--)
        keys[j+1]=keys[j];
    keys[i] = y->keys[t-1];
    n=n+1;
}

class BTreeNode
{
    int *keys
    int t;
    BTreeNode **child;
    int n;
    bool leaf;
    //function declarations
}
class BTree
{
    BTreeNode *root;
    int t;
    // functions
}
```


Anusree