11-11-2020

ADS-LAB 6

Nitish. N. B
1BM18CS063

## B-Tree Insertion:

```
insert (int k)
    if (root is NULL)
        root = new node (t);
        root -> keys[0] = k
        robt -> n = 1;
    else
        if (root->n == 2*t -1)
            s = new Node(t);
            s -> c[0] = root;
            s->splitchild (0, root);
            i=0
            if (s->keys[0] < k)
                i++;
            s -> c[i] ->insert NonFull (k)
            root = s
        else
            root ->insertNonFull (k)
```

```
Node  insertNonFull (int k)
    i = n - 1
    if (leaf is true)
        while (i >=0 && keys[i] > k)
            do  keys [i+1] = keys [i]
                i--;
        keys [i+1] = k
        n = n+1
```

```
class Node{
    int *keys;
    int t;
    Node **C;
    int n;
    bool leaf;
}
```

Nitish

```
else
    while (i>=0 && key[i] > k)
        do 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)

Node splitchild (int i, Node *y)
z = new Node(y->t)
z->n = t-1
for(int j=0; j<t-1; j++)
    z->key[j] = y->keys[j+t]
if (y->leaf is false)
    for(int j=0; j<t; j++)
        z->C[j] = y->C[j+t]
y->n = t-1
for (j=n; j>=i+1; j--)
    c[j+1] = C[j]
C[i+1] = z
for (j=n-1; j>=i; j--)
    keys[j+1] = keys[j]
key[i] = y->keys[t-1]
n = n+1
```

Nilab