J. Anukesh
1BM18CS038
11/11/2020

B-Tree

Write-up

```
void BTree::insert(int k)
{
    if(root == NULL)
    {
        root = new Node(t,true);
        root->key[0]=k;
        root->n=1;
    }
    else
    {
        if(root->n == 2*t-1)
        {
            Node *s = new Node(t,false);
            s->C[0]=root;
            s->splitchild(0,root);
            int i=0;
            if (s->key[0]<k)
                i++;
            s->C[i]->insertNonFull(k);
            root=s;
        }
        else
            root->insertNonFull(k);
    }
}

void Node::insertNonFull(int k)
{
    int i=n-1;
    if(l==true)          // l is leaf here
    {
        while(i>=0 && key[i]>k)
        {
            key[i+1]=key[i];
            i--;
        }
        key[i+1]=k;
        n=n+1;
    }
```

J. Anukesh
1BM18CS038
11/11/2020

```
else
{
    while (i>=0 && key[i]>k)
        i--;
    if (C[i+1]->n == 2*t-1)
    {
        split(child(i+1)C[i+1]);
        if (key[i+1]<k)
            i++;
    }
    C[i+1]->insertNonFull(k);
}
}
void Node::splitChild(int i, int *y)
{
    Node *z = new Node(y->t, y->l);
    z->n = t-1;
    for(int j=0; j<t-1; j++)
        z->key[j] = y->key[j+t];
    if(y->l == false)
    {
        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--)
        C[j+1] = C[j];
    C[i+1] = z;
    for(int j=n-1; j>=i; j--)
        key[j+1] = key[j];
    key[i] = y->key[t-1];
    n = n+1;
}
```