Prog-6 : B-Trees - Insertion

Shikha N
1BM18CS149
11-11-20

```
class BTreeNode
{
    int *key;
    int d;
    BTreeNode **child
    int n;
    bool leaf;
public:
    // constructor & function declarations
    friend class BTree;
};
class BTree {
    BTreeNode *root = NULL;
    int d;
    void insert(int k);
    void traverse(){
            if root != NULL
                root -> traverse();
    }
};
BTreeNode::BTreeNode(int dx, bool leafx)
{
    d = dx;
    leaf = leafx;
    keys = new int[2*d-1];
    child = new BTreeNode *[2*d];
    n = 0;
}
void BTree::insert(int k) {
    if root == NULL {
        root = new BTreeNode(d, true);
        root -> key[0] = k;
        root -> n = 1; }
```

(1)

```
else {
    if (root->n == 2*d-1)
    {
        BTreeNode *S = new BTreeNode(d, false);
        S->child[0] = root;
        S->splitchild(0, root);
        int i=0;
        if (S->key[0] < k)
            i++;
        S->child[i]->insertNonFull(k);
        root = S;
    }
    else {
        root->insertNonFull(k);
    }
}
}
Void BTreeNode::insertNonFull(int k) {
    int i = n-1;
    if (leaf == true) {
        while (i>=0 && key[i]<k) {
            key[i+1] = key[i]
            i--
        }
        key[i+1] = k
        n++;
    }
}
```

```
else {
while (i >= 0 && key[i] > k)
        i--;
if (child[i+1]->n == 2*d+1)
  { splitchild(i+1, child[i+1]);
    if(key[i+1] < k)
        i++;
  }
  child[i+1]-> insertNonFull(k);
 }
}
void BTreeNode :: splitchild(int i, BTreeNode *y)
{ BTreeNode *z = new BTreeNode(y->d, y->leaf)
   z->n = d-1;
for (int j=0; j<d-1; j++)
    z->key[j] = y->key[j+d];
if (y->leaf == false)
 { for (int j=0; j<d; j++)
     z->child[j] = y->child[j+d];
 }
 y->n = d-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--)
    key[j+1] = key[j]
key[i] = y->key[d-1]; n++; }
```


Shikha