**PROGRAM 7)** Write a program to implement insertion operation on a B-tree

CODE:

#include <bits/stdc++.h>

using namespace std;

class BTreeNode

{

int \*keys;

BTreeNode \*\*child;

int t;

int n;

bool leaf;

public:

BTreeNode(int t, bool leaf);

void traverse();

void insertNonFull(int k);

void splitChild(int i, BTreeNode \*y);

friend class BTree;

};

class BTree

{

BTreeNode \*root;

int t;

public:

BTree(int \_t){

root = NULL;

t = \_t;

}

void traverse()

{

if(root != NULL)

root->traverse();

}

void insert(int k);

};

BTreeNode::BTreeNode(int t1, bool leaf1)

{

t = t1;

leaf = leaf1;

keys = new int[2\*t-1];

child = new BTreeNode \*[2\*t];

n = 0;

}

void BTree::insert(int k)

{

if(root == NULL)

{

root = new BTreeNode(t, true);

root->keys[0] = k;

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 BTreeNode::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;

}

void BTreeNode::traverse()

{

//cout<<endl;

int i;

for(i=0; i<n; i++)

{

if(leaf == false)

child[i]->traverse();

cout<<keys[i]<<" ";

}

if(leaf == false)

child[i]->traverse();

//cout<<endl;

}

int main()

{

int d;

cout<<"Enter the degree: ";

cin>>d;

BTree t(d);

int n,k;

cout<<"Enter the no. of elements"<<endl;

cin>>n;

cout<<"Enter the keys"<<endl;

for(int i=0; i<n; i++)

{

cin>>k;

t.insert(k);

}

cout << "Traversal of tree constructed is\n";

t.traverse();

cout<<endl;

return 0;

}

OUTPUT:

