**NAME:SAYALI JIVAN CHAUDHARI**

**ROLL NO.:14**

**PRN NO.2023015400005055**

**1)implementation of b tree in cpp**

#include <bits/stdc++>

#include<iostream>

using namespace std;

class Tree {

int \*keys;

int t;

Tree \*\*C;

int n;

bool leaf\_node;

public:

Tree(int \_t, bool \_leaf\_node);

void insert\_more(int k);

void split\_child(int i, Tree \*y);

void print();

friend class BTree;

};

class BTree {

Tree \*root;

int t;

public:

BTree(int \_t) {

root = NULL;

t = \_t;

}

void print() {

if (root != NULL)

root->print();

}

void insert\_more(int k);

};

Tree::Tree(int t1, bool leaf\_node1) {

t = t1;

leaf\_node = leaf\_node1;

keys = new int[2 \* t – 1];

C = new Tree \*[2 \* t];

n = 0;

}

// print the Trees

void Tree::print() {

int i;

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

if (leaf\_node == false)

C[i]->print();

cout << ” “ << keys[i];

}

if (leaf\_node == false)

C[i]->print();

}

// Insert the Tree

void BTree::insert\_more(int k) {

if (root == NULL) {

root = new Tree(t, true);

root->keys[0] = k;

root->n = 1;

} else {

if (root->n == 2 \* t – 1) {

Tree \*s = new Tree(t, false);

s->C[0] = root;

s->split\_child(0, root);

int i = 0;

if (s->keys[0] < k)

i++;

s->C[i]-> insert\_more(k);

root = s;

} else

root-> insert\_more(k);

}

}

// Insert non full condition

void Tree:: insert\_more(int k) {

int i = n – 1;

if (leaf\_node == 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 (C[i + 1]->n == 2 \* t – 1) {

split\_child(i + 1, C[i + 1]);

if (keys[i + 1] < k)

i++;

}

C[i + 1]->insert\_more(k);

}

}

// split the child

void Tree::split\_child(int i, Tree \*y) {

Tree \*z = new Tree(y->t, y->leaf\_node);

z->n = t – 1;

for (int j = 0; j < t – 1; j++)

z->keys[j] = y->keys[j + t];

if (y->leaf\_node == 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–)

keys[j + 1] = keys[j];

keys[i] = y->keys[t – 1];

n = n + 1;

}

int main() {

BTree t(3);

t.insert\_more(11);

t.insert\_more(12);

t.insert\_more(13);

t.insert\_more(14);

t.insert\_more(15);

t.insert\_more(16);

t.insert\_more(17);

t.insert\_more(18);

t.insert\_more(19);

t.insert\_more(20);

cout << “The B-tree is: “;

t.print();

}