```
// ASSIGNMENT-6

#include <iostream>
#define MAX_VALUE 65536
using namespace std;
class N
{ // node declaration
public:
    int k;
    N *l, *r;
    bool leftTh, rightTh;
};
class ThreadedBinaryTree
{
private:
    N *root;

public:
    ThreadedBinaryTree()
    { // constructor to initialize the variables
        root = new N();
        root->r = root->l = root;
        root->leftTh = true;
        root->k = MAX_VALUE;
    }
    void insert(int key)
    {
        N *p = root;
        for (;;)
        {
            if (p->k < key)
    { // move to right thread
        if (p->rightTh)
            break;
        p = p->r;
    }
    else if (p->k > key)
    { // move to left thread
        if (p->leftTh)
            break;
        p = p->l;
    }
    else
    {
        return;
    }
}
N *temp = new N();
temp->k = key;
temp->rightTh = temp->leftTh = true;
if (p->k < key)
{
    temp->r = p->r;
    temp->l = p;
    p->r = temp;
    p->rightTh = false;
}
else
{
    temp->r = p;
    temp->l = p->l;
    p->l = temp;
```

```
                p->leftTh = false;
            }
        }
        void inorder()
        { // print the tree
            N *temp = root, *p;
            for (;;)
            {
                p = temp;
                temp = temp->r;
                if (!p->rightTh)
                {
                    while (!temp->leftTh)
                    {
                        temp = temp->l;
                    }
                }
                if (temp == root)
                    break;
                cout << temp->k << " ";
            }
            cout << endl;
        }
    };
    int main()
    {
        ThreadedBinaryTree tbt;
        cout << "Threaded Binary Tree\n";
        tbt.insert(56);
        tbt.insert(23);
        tbt.insert(89);
        tbt.insert(85);
        tbt.insert(20);
        tbt.insert(30);
        tbt.insert(12);
        tbt.inorder();
        cout << "\n";
    }
```

```
OUTPUT:
Threaded Binary Tree
12 20 23 30 56 85 89
```