#include <iostream>

using namespace std;

class TBTNode

{

public:

int data, Lbit, Rbit;

TBTNode \*left, \*right;

};

class TBTree

{

public:

TBTNode \*curr, \*temp, \*head, \*root;

TBTree()

{

root = NULL;

}

void create();

void inorder(TBTNode \*);

};

void TBTree :: create()

{

char s;

int flag;

TBTNode \*node, \*temp;

head = new TBTNode;

head->left=head;

head->right=head;

head->Rbit=head -> Lbit =1;

root = new TBTNode;

cout<<endl<<"\n Enter data for root = ";

cin >> root -> data;

root->left=head;

root->right=head;

head->left = root;

root->Lbit=root->Rbit=1;

do

{

node = new TBTNode;

cout<<"Enter next data = ";

cin>> node->data;

node->Lbit=node->Rbit=1;

temp = root;

while(1)

{

if(node->data<temp->data)

{

if(temp->Lbit==1)

{

node->left=temp->left;

node->right=temp;

temp->Lbit=0;

temp->left=node;

break;

}

else

temp=temp->left;

}

else{

if(temp->Rbit==1)

{

node->left=temp;

node->right=temp->right;

temp->right = node;

temp->Rbit = 0;

break;

}

else

temp=temp->right;

}

}

cout<<"\n\nDo you want to add more data ?[y/n]";

cin>>s;

}

while(s=='y'|| s=='Y');

}

void TBTree::inorder(TBTNode \*)

{

TBTNode \*temp;

temp=root;

int flag=0;

if(root==NULL)

{

cout<<"Tree not present";

}

else

{

while(temp!=head)

{

if (temp->Lbit==0 && flag==0)

{

temp=temp->left;

}

else

{

cout<<temp->data<<" ";

if(temp->Rbit==0)

{

temp=temp->right;

flag=0;

}

else

{

temp=temp->right;

flag=1;

}

}

}

}

}

int main()

{

TBTree t;

cout<<"\n\t\*\*\*\* Threaded Binary Tree \*\*\*\*";

t.create();

cout<<"\n\nInorder Traversal: ";

t.inorder(t.root);

return 0;

}