**ASSIGNMENT 2**

**AIM:**

Construct a threaded binary search tree by inserting values in the given order and traverse it in inorder traversal using threads.

**CODE:**

#include<iostream>

using namespace std;

class ttree

{

private:

struct thtree

{

int left;

thtree \*leftchild;

int data;

thtree \*rightchild;

int right;

}\*th\_head;

public:

ttree();

void insert(int num);

void inorder();

};

ttree::ttree()

{

th\_head=NULL;

}

void ttree::insert(int num)

{

thtree \*head=th\_head,\*p,\*z;

z=new thtree;

z->left=true;

z->data=num;

z->right=true;

if(th\_head==NULL)

{

head=new thtree;

head->left=false;

head->leftchild=z;

head->data=-9999;

head->rightchild=head;

head->right=false;

th\_head=head;

z->leftchild=head;

z->rightchild=head;

}

else

{

p=head->leftchild;

while(p!=head)

{

if(p->data > num)

{

if(p->left!=true)

p=p->leftchild;

else

{

z->leftchild=p->leftchild;

p->leftchild=z;

p->left=false;

z->right=true;

z->rightchild=p;

return;

}

}

else

{

if(p->data < num)

{

if(p->right!=true)

p=p->rightchild;

else

{

z->rightchild=p->rightchild;

p->rightchild=z;

p->right=false;

z->left=true;

z->leftchild=p;

return;

}

}

}

}

}

}

void ttree::inorder()

{

thtree \*a;

a=th\_head->leftchild;

while(a!=th\_head)

{

while(a->left==false)

a=a->leftchild;

cout<<a->data<<"\t";

while(a->right==true)

{

a=a->rightchild;

if(a==th\_head)

break;

cout<<a->data<<"\t";

}

a=a->rightchild;

}

}

int main()

{

ttree th;

int n,e;

cout<<"Enter no. of elements: ";

cin>>n;

cout<<"\nEnter elements: ";

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

{

cin>>e;

th.insert(e);

}

cout<<"\n\n\tThreaded binary tree in inorder: ";

th.inorder();

}

