/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Threaded Binary tree

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

using namespace std;

class node

{

public:

int data;

int lth,rth;

node \*left,\*right;

};

class thread

{

public:

node \*dummy;

node \*New,\*root,\*temp,\*parent;

thread();

void create();

void insert(node \*,node \*);

void display\_inorder(node \*,node \*);

void display\_preorder(node \*,node \*);

};

thread::thread()

{

root=NULL;

}

void thread::create()

{

New=new node;

New->left=NULL;

New->right=NULL;

New->lth=0;

New->rth=0;

cout<<"\n Enter The Element: ";

cin>>New->data;

if(root==NULL)

{

root=New;

dummy=new node;

dummy->data=-999;

dummy->left=root;

root->left=dummy;

root->right=dummy;

}

else

insert(root,New);

}

void thread::insert(node \*root,node \*New)

{

if(New->data<root->data)

{

if(root->lth==0)

{

New->left=root->left;

New->right=root;

root->left=New;

root->lth=1;

}

else

insert(root->left,New);

}

if(New->data>root->data)

{

if(root->rth==0)

{

New->right=root->right;

New->left=root;

root->rth=1;

root->right=New;

}

else

insert(root->right,New);

}

}

void thread::display\_inorder(node \* r,node \*d)

{

temp=r;

while(temp!=d)

{

while(temp->lth==1)

{

temp=temp->left;

}

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

while(temp->rth==0)

{

temp=temp->right;

if(temp==d)

return;

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

}

temp=temp->right;

}

}

void thread::display\_preorder(node \*temp,node \*dummy)

{

int flag=0;

while(temp!=dummy)

{

if(flag==0)

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

if((temp->lth==1)&&(flag==0))

{

temp=temp->left;

}

else

{

while(1)

{

if(temp->rth==1)

{

flag=0;

temp=temp->right;

break;

}

else

{

if(temp!=dummy)

{

flag=1;

temp=temp->right;

break;

}

}

}

}

}

}

int main()

{

int choice;

thread th;

do

{

cout<<"\nProgram For Threaded Binary Tree\n";

cout<<"\n1)Create \n2)Display Inorder\n3)Display Preorder\n4)Exit";

cout<<"\nEnter Your Choice: ";

cin>>choice;

switch(choice)

{

case 1:

th.create();

break;

case 2:

th.display\_inorder(th.root,th.dummy);

break;

case 3:

th.display\_preorder(th.root,th.dummy);

break;

}

}while(choice!=4);

return 0;

}