ASSIGNMENT 10

Name : AARYAN BAIRAGI

Roll no: 27002

Class : SE IT-B

Problem Statement :Implement In-order Threaded Binary Tree and traverse it in In-order and Pre-order.\*/

#include<iostream>

using namespace std;

class node

{

    public:

        int data;

        int lth,rth;

        node \*left,\*right;

};

class thread

{

    private:

    node \*dummy;

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

    public:

    thread();

    void create();

    void insert(node \*,node \*);

    void display();

    void find();

    void delet();

    void del(node \*root,node \*dummy,int key);

};

/\*

---------the constructor defined--------

\*/

thread::thread()

{

    root=NULL;

}

/\*

----------the create function-------------

\*/

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);

}

/\*

-------This function is for creating a binary search tree

\*/

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);

    }

}

/\*

--------The display function---------

\*/

void thread::display()

{

    void inorder(node\*,node \*dummy);

    void preorder(node\*,node \*dummy);

    if(root==NULL)

        cout<<"Tree is not created";

    else

        {

            cout<<"\nInorder Traversal...";

            inorder(root,dummy);

            cout<<"\nPreorder Traversal...";

            preorder(root,dummy);

        }

}

/\*

----The inorder function---------

\*/

void inorder(node \*temp,node \*dummy)

{

    while(temp!=dummy)

    {

        while(temp->lth==1)

        temp=temp->left;

    cout<<" "<<temp->data;

    while(temp->rth==0)

    {

        temp=temp->right;

        if(temp==dummy)

        return;

        cout<<" "<<temp->data;

    }

    temp=temp->right;

    }

}

/\*

----the preorder function---------

\*/

void 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;

    char ans='N';

    thread th;

    do

    {

        cout<<"\n\t Program For Threaded Binary Tree";

        cout<<"\n1.Create \n2.Display";

        cin>>choice;

        switch(choice)

        {

            case 1:

            do

            {

                th.create();

                cout<<"\n Do u want to enter more elements?(y/n)";

                cin>>ans;

            }while(ans=='y');

            break;

            case 2:th.display();

            break;

        }

        cout<<"\n\nWant To See Main Menu?(y/n)";

        cin>>ans;

    }while(ans=='y');

    return 0;

}