/\*Convert given binary tree into threaded binary tree. Analyse time and space complexity of algorithm.\*/

#include<iostream>

using namespace std;

class node

{

int data;

int lbit,rbit;

node\* left,\* right;

public:

/\*node()

{

left=right=NULL;

lbit=rbit=0;

}\*/

node(int d)

{

data=d;

left=right=NULL;

lbit=rbit=0;

}

friend class TBT;

};

class TBT

{

node\* root;

public:

TBT()

{

root->data=9999;

root->left=root->right=root;

root->lbit=0;

root->rbit=1;

}

void create();

void inorder();

node\* inorder\_successor(node\*);

void insert(int);

};

void TBT::create()

{

int n;

if(root->left==root && root->right==root)

{

cout<<"Enter number of roots: ";

cin>>n;

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

{

int value;

cout<<"\n Enter data: ";

cin>>value;

insert(value);

}

}

else

{

cout<<"\n Tree is already created.";

}

};

void TBT::insert(int value)

{

if(root->left==root && root->right==root)

{

node\* p=new node(value);

p->left=root->left;

p->right=root->right;

root->left=p;

root->lbit=1;

}

else

{

node\* current=root->left;

node\* p=new node(value);

while(1)

{

if(current->data>value)

{

if(current->lbit==0)

{

p->left=current->left;

p->right=current;

current->left=p;

current->lbit=1;

return;

}

else

{

current=current->left;

}

}

else

{

if(current->data<value)

{

if(current->rbit==0)

{

p->right=current->right;

p->left=current;

current->right=p;

current->rbit=1;

return;

}

else

{

current=current->right;

}

}

}

}

}

};

void TBT::inorder()

{

node\* temp=root->left;

while (temp->lbit==1)

{

temp=temp->left;

}

while(temp!=root)

{

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

temp=inorder\_successor(temp);

}

};

node\* TBT::inorder\_successor(node\* temp)

{

if(temp->rbit==0)

{

return temp->right;

}

else

{

temp=temp->right;

while(temp->lbit==1)

{

temp=temp->left;

}

return temp;

}

}

int main()

{

int choice;

TBT T1;

do

{

cout<<"\n\n1. Create TBT";

cout<<"\n2. Inorder";

cout<<"\n3. EXIT \n";

cout<<"\n Enter your choice: ";

cin>>choice;

cout<<"\n";

switch(choice)

{

case 1:

T1.create();

break;

case 2:

T1.inorder();

break;

case 3:

break;

}

}while(choice!=3);

return 0;

};

**Output**:

Inorder traversal of created threaded tree is

4 2 5 1 6 3 7

--------------------------------

Process exited after 0.1821 seconds with return value 0

Press any key to continue . . .