#include <iostream>

#include<string.h>

using namespace std;

struct BST

{

int item;

struct BST \*left;

struct BST \*right;

};

typedef struct BST node;

node \*root=NULL, \*temp;

int main()

{

int choice;

void insert();

void inorder(node \*root);// L N R

void preorder(); //N L R

void postorder(); //L R N

void findmin(node \*);

void findmax(node \*);

do{

cout<<"Enter your choice ";

cin>>choice;

switch(choice)

{

case 1: insert();break;

case 2: inorder(root); break;

case 3: preorder(root); break;

case 4: postorder(root); break;

//case 5: findmin(root); break;//try at home

//case 6: findmax(root); break;//try at home

case 4: break;

}

}while(choice!=);

return 0;

}

void insert()

{

void create();// simply create a node

void search(node \*root);

create(); //call create function()

if(root==NULL) // root--> |2000|10|3000|

root=temp;

else // t---> |5000|8|4000| |NULL|20|NULL|

search(root);

} // |NULL|7|NULL| |NULL|9|NULL|

void create()

{

int ele;

cout<<"Enter the element";

cin>>ele;

temp=new node;//allocate the memory;

temp->left=NULL;

temp->item=ele;

temp->right=NULL;

}

void search(node \*t)

{

if(temp->item < t->item && t->left==NULL) //case 1

t->left=temp; //as a leaf node

if(temp->item < t->item && t->left!=NULL)//case 2

search(t->left);

if(temp->item > t->item && t->right==NULL)//case 3

t->right=temp; //as a leaf node

if(temp->item > t->item && t->right!=NULL)//case 4

search(t->right);

}

void inorder(node \*t)

{

if(t->left!=NULL) //LEFT

inorder(t->left);

cout<<" "<<t->item; //NODE or print

if(t->right!=NULL)//RIGHT

inorder(t->right);

}

void preorder(node \*t)

{

cout<<" "<<t->item; //NODE

if(t->left!=NULL) //LEFT

inorder(t->left);

if(t->right!=NULL)//RIGHT

inorder(t->right);

}

void postorder(node \*t)

{

if(t->left!=NULL) //LEFT

inorder(t->left);

//NODE

if(t->right!=NULL)//RIGHT

inorder(t->right);

cout<<" "<<t->item; // NODE

}