**14. Write a C++ program that uses functions to perform the following:**

**a) Create a binary search tree of integers.**

**b) Traverse the above Binary search tree non recursively in inorder.**

#include<iostream>

using namespace std;

class node

{

public:

class node \*left;

class node \*right;

int data;

};

class tree: public node

{

public:

int stk[50],top;

node \*root;

tree()

{

root=NULL;

top=0;

}

void insert(int ch)

{

node \*temp,\*temp1;

if(root== NULL)

{

root=new node;

root->data=ch;

root->left=NULL;

root->right=NULL;

return;

}

temp1=new node;

temp1->data=ch;

temp1->right=temp1->left=NULL;

temp=search(root,ch);

if(temp->data>ch)

temp->left=temp1;

else

temp->right=temp1;

}

node \*search(node \*temp,int ch)

{

if(root== NULL)

{

cout <<"no node present";

return NULL;

}

if(temp->left==NULL && temp->right== NULL)

return temp;

if(temp->data>ch)

{ if(temp->left==NULL)

return temp;

search(temp->left,ch);}

else

{ if(temp->right==NULL)

return temp;

search(temp->right,ch);

}

}

void display(node \*temp)

{

if(temp==NULL)

return ;

display(temp->left);

cout<<temp->data;

display(temp->right);

}

void inorder( node \*root)

{

node \*p;

p=root;

top=0;

do

{

while(p!=NULL)

{

stk[top]=p->data;

top++;

p=p->left;

}

if(top>0)

{

p=pop(root);

cout << p->data;

p=p->right;

}

}while(top!=0 || p!=NULL);

}

node \* pop(node \*p)

{

int ch;

ch=stk[top-1];

if(p->data==ch)

{

top--;

return p;

}

if(p->data>ch)

pop(p->left);

else

pop(p->right);

}

};

int main()

{

tree t1;

int ch,n,i;

while(1)

{

cout <<"\n1.INSERT\n2.DISPLAY\n3.INORDER TRAVERSAL\n4.EXIT\nEnter your choice:";

cin >> ch;

switch(ch)

{

case 1: cout <<"enter no of elements to insert:";

cin >> n;

for(i=1;i<=n;i++)

{ cin >> ch;

t1.insert(ch);

}

break;

case 2: t1.display(t1.root);break;

case 3: t1.inorder(t1.root); break;

case 4: exit(1);

}

}

return 0;

}

**Output:-**

