#**include**<bits/stdc++.h>

**using** **namespace** std;

**class** node{

**private**:

**long** **long** **int** num;

node \*right, \*left;

**public**:

node(**int** key):num(key){

right=left=nullptr;

}

**friend** **class** bst;

};

**class** bst{

**private**:

node \*root;

**void** **add**(**long** **long** **int** key,node \*cur){

**if**(root==nullptr){

node \*child=**new** node(key);

root=child;

}

**else**{

**if**(key>=cur->num){

**if**(**static\_cast**<**bool**>(cur->right))

add(key,cur->right);

**else**{

node \*child=**new** node(key);

cur->right=child;

}

}

**else**{

**if**(**static\_cast**<**bool**>(cur->left))

add(key,cur->left);

**else**{

node \*child=**new** node(key);

cur->left=child;

}

}

}

}

**void** **destruct**(node \*cur){

**if**(**static\_cast**<**bool**>(cur->left))

destruct(cur->left);

**if**(**static\_cast**<**bool**>(cur->right))

destruct(cur->right);

**delete** cur->left;

**delete** cur->right;

cur->left=nullptr;

cur->right=nullptr;

}

**bool** **check\_null**(**void**){

**if**(root==nullptr)

**return** true;

**else**

**return** false;}

**public**:

bst(){

root=nullptr;

}

bst(vector<**long** **long** **int**> vec):root(nullptr){

**for**(**const** **auto**& i:vec){

add(i,root);

}

}

~bst(){

**if**(root!=nullptr)

destruct(root);

root=nullptr;

}

**void** **push**(**int** key){

add(key,root);

}

**void** **in**(node \*cur){

**if**(**static\_cast**<**bool**>(cur->left))

in(cur->left);

cout<<cur->num<<' ';

**if**(**static\_cast**<**bool**>(cur->right))

in(cur->right);

}

**void** **pre**(node\*cur){

cout<<cur->num<<' ';

**if**(**static\_cast**<**bool**>(cur->left))

pre(cur->left);

**if**(**static\_cast**<**bool**>(cur->right))

pre(cur->right);

}

**void** **post**(node\*cur){

**if**(**static\_cast**<**bool**>(cur->left))

post(cur->left);

**if**(**static\_cast**<**bool**>(cur->right))

post(cur->right);

cout<<cur->num<<' ';

}

**void** **print**(string a){

**if**(a=="pre")

pre(root);

**else** **if**(a=="in")

in(root);

**else**

post(root);

}

**void** **clean**(**void**){

**if**(root!=nullptr)

destruct(root);

root=nullptr;

}

**bool** **empty**(**void**){

**return** check\_null();

}

};

**int** **main**(){

cin.tie(nullptr);

ios::sync\_with\_stdio(0);

string s;

**while**(getline(cin,s))

{

bst tree;

string a;

cin>>a;

stringstream ss;

ss<<s;

**long** **long** **int** tmp;

**while**(ss>>tmp)

tree.push(tmp);

tree.print(a);

tree.clean();

}

**return** 0;

}