#include<bits/stdc++.h>

using namespace std;

class node{

public:

int data;

node\* right;

node\* left;

node(int d){

data = d;

right = NULL;

left = NULL;

}

};

node\* insertnode(){

int value;

cin>>value;

if(value==-1){

return NULL;

}

node \*root = new node(value);

root->left = insertnode();

root->right = insertnode();

return root;

}

void inorder(node \*root){

if(root==NULL){

return;

}

else{

inorder(root->left);

cout<<root->data<<" ";

inorder(root->right);

}

}

void preorder(node\* root){

if(root==NULL){

return;

}

else{

cout<<root->data<<" ";

preorder(root->left);

preorder(root->right);

}

}

void postorder(node\* root){

if(root==NULL){

return;

}

else{

postorder(root->left);

postorder(root->right);

cout<<root->data<<" ";

}

}

void bfs(node\* root){

if(root==NULL){

return;

}

queue<node\*> q;

q.push(root);

while(!q.empty()){

node\* front = q.front();

cout<<front->data<<" ";

q.pop();

if(front->left!=NULL){

q.push(front->left);

}

if(front->right!=NULL){

q.push(front->right);

}

}

cout<<endl;

}

int height(node\* root){

if(root==NULL){

return 0;

}

int l = height(root->left);

int r = height(root->right);

int ans = 1 + max(l,r);

return ans;

}

int size(node \*root){

if(root==NULL){

return 0;

}

return 1+size(root->left)+size(root->right);

}

int sum(node\* root){

if(root==NULL){

return 0;

}

return root->data+sum(root->left)+sum(root->right);

}

int maximum(node \*root){

if(root==NULL){

return INT\_MIN;

}

int large = root->data;

int lr = max(maximum(root->left),maximum(root->right));

large = max(large,lr);

return large;

}

int main(){

//1 2 3 -1 4 -1 -1 -1 5 -1 -1

node\* root = insertnode();

inorder(root);

cout<<endl;

preorder(root);

cout<<endl;

postorder(root);

cout<<endl;

bfs(root);

cout<<size(root)<<endl;

cout<<sum(root)<<endl;

cout<<maximum(root)<<endl;

cout<<height(root)<<endl;

}