#include<iostream>

#include<cmath>

#include<cstring>

using namespace std;

class Node {

public:

char data[4];

//用于实现树

Node\* left;

Node\* right;

//用于实现队列

Node\* next;

Node() {

left = NULL;

right = NULL;

next = NULL;

}

Node(char\* x) {

strcpy(data, x);

left = NULL;

right = NULL;

next = NULL;

}

};

class ListQueue {

public:

Node\* front1;

Node\* front2;

Node\* rear;

ListQueue() {

front1=front2= rear = NULL;

}

void push(char\* c) {

Node\* tmp = new Node(c);

if(front1==NULL){

front1=front2= tmp;

rear = tmp;

}

else {

rear->next = tmp;

tmp->next = NULL;

rear = tmp;

}

}

void pop() {

if (front1== NULL)return;

front1= front1->next;

}

};

class Tree {

public:

ListQueue L;

bool t1 ;

bool t2 ;

Node\* root;

Tree() {

root = NULL;

t1 = false;

t2 = false;

}

Node\* & find(Node\* n) {

if (n->left == NULL)return n->left;

else {

if (n->right == NULL) return n->right;

else find(n->next);

}

}

bool judge(Node\* n) {

if ( !strcmp(n->data , "#" )&& !t1) {

t1 = true;

}

if (strcmp(n->data, "#") && t1) {

t2 = true;

}

if (t1 && t2) {

return false;

}

if (n->next == NULL)return true;

judge(n->next);

}

void insert(char\* n) {

Node\* tmp = new Node(n);

if (root == NULL) {

root = tmp;

L.push(n);

}

else {

find(L.front1) = tmp;

L.push(tmp->data);

if (L.front1->right != NULL) {

L.pop();

}

}

}

int height(char\* c) {

int count = 1;

Node\* tmp = new Node();

tmp = L.front2;

while (true) {

if (strcmp(tmp->data, c)) {

tmp = tmp->next;

count++;

}

else {

break;

}

}

int height = (log(count) / log(2)) + 1;

return height;

}

};

int main() {

Tree t;

Tree r;

char c[4];

cout << "请输入元素" << endl;

while (cin >> c) {

t.insert(c);

}

cin.clear();

cout << "该树是否为完全二叉树？" << endl;

if (t.judge(t.L.front2))

cout << "是" << endl;

else {

cout << "不是" << endl;

cout << "下面我要将其转换成完全二叉树形式输出：" << endl;

Node\* m = new Node();

m = t.L.front2;

while (true) {

if (m == NULL) {

break;

}

if (strcmp(m->data, "#")) {

r.insert(m->data);

}

m = m->next;

}

}

Node\* tmp = new Node();

tmp = r.L.front2;

int n = 1;

for (;;) {

if (tmp == NULL) {

break;

}

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

if (tmp == NULL) {

break;

}

cout << tmp->data << " ";

tmp = tmp->next;

}

cout << endl;

n = n \* 2;

}

cout << endl;

cout << "请输入想查询的节点值" << endl;

cin >> c;

cout << t.height(c);

return 0;

}