代码：

#include <iostream>

using namespace std;

class Bitnode {

private:

char data;

Bitnode\* right;

Bitnode\* left;

public:

Bitnode(char data1) {

data = data1;

right = nullptr;

left = nullptr;

}

Bitnode\* getleft() { return left; }

Bitnode\* getright() { return right; }

char getdata() { return data; }

void cleft(Bitnode\* left1) { left=left1; }

void cright(Bitnode\* right1) { right=right1; }

void cdata(char data1) { data=data1; }

friend class Bitree;

};

class Bitree {

private:

Bitnode\* root;

public:

void Deletetree (Bitnode\* &T) {

if (T) {

Deletetree(T->left);

Deletetree(T->right);

delete T;

}

}

~Bitree() {

Deletetree(root);

}

void CreateBitree(Bitnode\*& T) {

char ch;

cin >> ch;

if (ch == '@') {

T= nullptr;

}else {

T = new Bitnode(ch);

CreateBitree(T->left);

CreateBitree(T->right);

}

}

void Preorder(Bitnode \*& T) {

if (T) {

cout << T->data;

Preorder(T->left);

Preorder(T->right);

}

}

void Inorder(Bitnode\*& T) {

if (T) {

Inorder(T->left);

cout << T->data;

Inorder(T->right);

}

}

void Postorder(Bitnode\* &T) {

if (T) {

Postorder(T->left);

Postorder(T->right);

cout << T->data;

}

}

int Count1(Bitnode\*& T) {

int n=0;

if (T) {

Count1(T->left);

Count1(T->right);

if (T->left && !T->right || !T->left && T->right)n++;

}

return n;

}

void Exchange(Bitnode\*& T) {

if (T) {

Exchange(T->left);

Exchange(T->right);

Bitnode\* temp=T->left;

T->left = T->right;

T->right = temp;

}

}

void Longest(Bitnode\*& T,char\*& Max,char\*& Temp,int& max,int temp) {

if (T) {

if (T->left || T->right) {

Temp[temp] = T->data;

temp++;

}

Longest(T->left, Max, Temp, max, temp);

Longest(T->right, Max, Temp, max, temp);

if (!T->left && !T->right) {

if (temp > max) {

Temp[temp] = T->data;

temp++;

max = temp;

for (int i = 0; i < max; i++) {

Max[i] = Temp[i];

}

}

}

}

}

void CreateBitree() {

CreateBitree(root);

}

void Preorder() {

Preorder(root);

}

void Inorder() {

Inorder(root);

}

void Postorder() {

Postorder(root);

}

void Count1() {

cout << Count1(root);

}

void Exchange() {

Exchange(root);

}

void Longest(){

char\* Max = new char[10];

char\* Temp=new char[10];

int max = 0,temp = 0;

Longest(root, Max, Temp, max, temp);

cout << max<<endl;

for (int i = 0; i < max; i++) {

cout << Max[i];

}

cout << endl;

delete[]Max;

delete[]Temp;

}

};

int main()

{

Bitree T;

T.CreateBitree();//前序创造

T.Preorder();//前序遍历

cout << endl;

T.Exchange();//交换左右

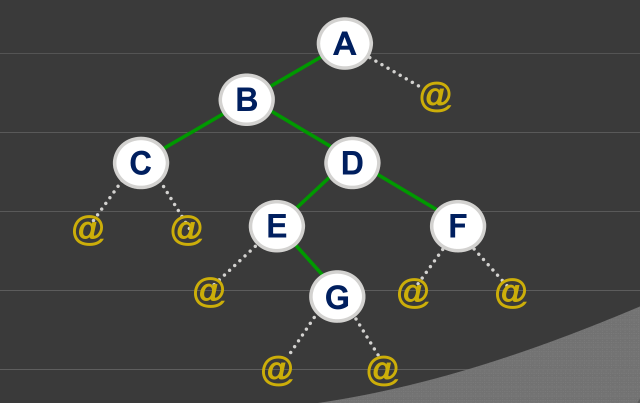
T.Preorder();//前序遍历

cout << endl;

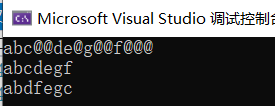
T.Longest();//遍历最长链

}

结果输入实例：abc@@de@g@@f@@@



倒序



遍历最长链

