实验四 二叉树的基本操作

# include <iostream>

using namespace std;

typedef struct BiTNode

{

char data;

struct BiTNode \*lchild,\*rchild;

}BiTNode,\*BiTree;

void createBiTree(BiTree \*B)

{

char c;

c=getchar();

if(c=='#')

\*B=NULL;

else

{

\*B=(BiTree)malloc(sizeof(BiTNode));

(\*B)->data=c;

(\*B)->lchild=NULL;

(\*B)->rchild=NULL;

createBiTree(&(\*B)->lchild);

createBiTree(&(\*B)->rchild);

}

}

void Preorder(BiTree B)

{

if(B==NULL)

return;

cout<<B->data<<" ";

Preorder(B->lchild);

Preorder(B->rchild);

}

void Inorder(BiTree B)

{

if(B==NULL)

return;

Inorder(B->lchild);

cout<<B->data<<" ";

Inorder(B->rchild);

}

void Postorder(BiTree B)

{

if(B==NULL)

return;

Postorder(B->lchild);

Postorder(B->rchild);

cout<<B->data<<" ";

}

int Countleaf(BiTree B,int &i)

{

if(B)

{

if((B->lchild==NULL)&&(B->rchild==NULL))

i=i+1;

Countleaf(B->lchild,i);

Countleaf(B->rchild,i);

}

return i;

}

int Depth(BiTree B)

{

int deep=0;

int depthleft,depthright;

if(B==NULL)

return deep;

else

{

depthleft=Depth(B->lchild);

depthright=Depth(B->rchild);

if(depthleft>depthright)

deep=depthleft+1;

else

deep=depthright+1;

}

return deep;

}

void CopyTree(BiTNode \*B,BiTNode \*&newB)

{

if(B==NULL)

newB=NULL;

else

{

newB= new BiTNode;

newB->data=B->data;

CopyTree(B->lchild,newB->lchild);

CopyTree(B->rchild,newB->rchild);

}

}

void Exchange(BiTree B)

{

BiTNode \*temp;

if(B==NULL)

return;

temp=B->lchild;

B->lchild=B->rchild;

B->rchild=temp;

Exchange(B->lchild);

Exchange(B->rchild);

}

void main()

{

BiTree B,A;

int i=0;

cout<<"输入一个二叉树"<<endl;

createBiTree(&B);

cout<<"先序遍历:";

Preorder(B);

cout<<endl;

cout<<"中序遍历:";

Inorder(B);

cout<<endl;

cout<<"后序遍历:";

Postorder(B);

cout<<endl;

cout<<"叶子结点:"<<Countleaf(B,i)<<endl;

cout<<"二叉树的深度:"<<Depth(B)<<endl;

CopyTree(B,A);

cout<<"复制后新二叉树先序序列:";

Preorder(A);

cout<<endl;

cout<<"交换二叉树的左右子树:";

Exchange(B);

Preorder(B);

cout<<endl;

}