**数据结构**

**——实验报告书**



姓名：樊雨晨

学号：201632110111

班级：软件工程(中外)16 班

学院：数理与信息工程学院

日期：2017年05月10日

# **实验四 由先序遍历和中序遍历序列建立二叉树**

## **问题**

按先序顺序和中序顺序输入二叉树的2个遍历序列，采用二叉链表建立该二叉树并用后序遍历顺序输出该二叉树的后序遍历序列。

## **实验思路**

1.首先得到二叉树，然后进行遍历。

## **实验步骤**

1.首先对先序序列和中序序列进行分解，前序序列可得根节点为第一个字符，找到这个字符在中序序列中的位置，可以得到左边的序列为该根节点的左子树，右边的序列为右子树，可以找到接下来的前序序列和中序序列。然后进行递归重复上述步骤。建立二叉树

2.将二叉树采用后序遍历，递归调用。

## **代码**

#**include**<stdio.h>

#**include**<stdlib.h>

#**include**<iostream>

#**include**<queue>

#**include**<stack>

#**include**<string.h>

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

#**define** Number 30

**typedef** **struct** BiTNode

{

**char** data;

**struct** BiTNode \*lchild, \*rchild;

}BiTNode,\*BiTree;

**void** **CreateBiTree**(BiTNode \*\*root,**char** \*before, **char** \*middle, **int** &L)

{

**int** i=-1,j,z,ll,rl;

**char** ltempbefore[Number];

**char** ltempmiddle[Number];

**char** rtempbefore[Number];

**char** rtempmiddle[Number];

**if**(L==0)

{

\*root=NULL;

**return**;

}

\*root = (BiTNode \*)malloc(**sizeof**(BiTNode));

(\*root)->data = \*before;

**for**(i=0;;i++)

{

**if**(\*(middle+i)==\*before)

**break**;

**else**

**continue**;

}

**for**(z=0;z<i;z++)

{

ltempbefore[z]=before[1+z];

ltempmiddle[z]=middle[z];

}

ltempbefore[z]='\0';

ltempmiddle[z]='\0';

**for**(i,j=0;i+1<L;i++,j++)

{

rtempbefore[j]=before[i+1];

rtempmiddle[j]=middle[i+1];

}

rtempbefore[j]='\0';

rtempmiddle[j]='\0';

ll=strlen(ltempbefore);

rl=strlen(rtempbefore);

CreateBiTree(&((\*root)->lchild),ltempbefore,ltempmiddle,ll);

CreateBiTree(&((\*root)->rchild),rtempbefore,rtempmiddle,rl);

}

**void** **PostOrder**(BiTNode \*root)

{

**if**(root==NULL)

**return** ;

PostOrder(root->lchild);

PostOrder(root->rchild);

printf("%c",root->data);

}

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

{

**int** l,i;

BiTNode \*root=NULL;

**char** before[Number];

**char** middle[Number];

**char** after[Number];

**while**(scanf("%s",before)!=EOF)

{

scanf("%s",middle);

l=strlen(before);

CreateBiTree(&root,before,middle,l);

PostOrder(root);

}

}

## **测试**

一开始在建立二叉树的过程中，始终不能够很好的判定是否存在子树，就是return的条件，从循环变量为0到最后新的序列的长度为0，终于成功得到了return的条件，并且在return时候需要将当前的结点变为空值。

## **结果**



## **总结**

1. 这次将很长的时间花在了调试上面，感觉还是有很多的收获的，可以仔仔细细的完整走一遍程序的过程，感觉对于程序的每一个步骤都有很深的了解。
2. 其次是在字符串的使用时候，如果要确定长度，可以在长度字符最后强制赋值空字符。
3. 在循环过程中，若为for(i=0,i<l,i++)，在最后判断出是否跳出循环时，i=l，所以结束了循环。