|  |
| --- |
| 哈尔滨工业大学(深圳) |
| **《数据结构》实验报告** |
|  |
| 实验三  树形结构及其应用  学 院: 计算机科学与技术   |  |  | | --- | --- | | 姓 名: | 朱海峰 | | 学 号: | 190110716 | | 专 业: | 计算机科学与技术 | | 日 期: | 2020-04-25 | |

# 问题分析

问题：

1、根据先序序列建立二叉树（其中#表示空），并输出中序和后序遍历。

2、根据后序和中序遍历序列构造二叉树，并输出先序遍历。

3、求给定二叉树叶子节点数和宽度。

4、判断两给定二叉树是否等价。

分析：

1. 逐一扫描先序序列，按照先序规则，对非空节点分配内存，建立节点。
2. 由后序序列找到根节点，在中序序列分开左右子树，再对左右子树递归执行此操作，直到二叉树完全建立。
3. 叶子节点数为左右子树叶子节点之和，递归地求左右子树叶子节点并相加即可；宽度则用层序遍历，将每一层的节点数记录，最大值即宽度。
4. 判断是否数据相同，左子树右子树均相同，也是递归用到此操作。

# 二、详细设计

## 2.1 设计思想

主要设计思想是递归,下文加粗表示。

(1)由先序序列建立二叉树：

1.建立根节点；

**2.先序建立左子树；**

**3.先序建立右子树。**

另外遇到#赋值空指针，此处不再递归。

(2)由后序和中序序列构造二叉树：

1.在后序序列最后一位确立根节点；

2.在中序序列找到根节点位置，分开左右子树；

**3.以左子树对应后序和中序序列构造二叉树；**

**4.以右子树对应后序和中序序列构造二叉树。**

另外考虑： 1)左右子树为空的情况不再递归；

2)未找到根节点，提示构造失败，退出程序；

3)中序后序长度不等，提示构造失败，退出程序；

4)序列长度为1，构造叶子节点即可。

(3)求二叉树叶子节点数：

**1.求左子树叶子节点数；**

**2.求右子树叶子节点数；**

3.返回两数之和。

其中： 1)空节点不含叶子；

2)没有子树的非空节点含叶子数为1。

(4)求二叉树宽度：

1.根节点入队，计数；

2.队头改为该层首节点，该层节点的子一层节点入队，计数，下一层；

3.重复操作2，直到队中没有节点

4.比较得出每层节点数最大值，即为宽度。

(5)判断两二叉树是否等价：

1.判断根节点数据是否相等，左右指针是否为空的情况相同与否；

**2.判断左子树是否等价；**

**3.判断右子树是否等价。**

其中，若1中有一个结果为否，得出不等价。

## 2.2 存储结构及操作

(1)存储结构

程序主要用到存储结构是链式二叉树。此外，在求宽度的时层序遍历二叉树会用到数组形式的队列。

(2)涉及的操作

BiTree createBiTree()

//函数功能建立二叉树

//返回：根节点

void preOrderTraverse(BiTree p)

//函数功能：前序遍历

void inOrderTraverse(BiTree p)

//函数功能：中序遍历

void postOrderTraverse(BiTree p)

//函数功能：后序遍历

BiTree PostInCreate(char post[],int s1,int e1,char in[],int s2,int e2)

//函数功能：根据后序与中序序列构造二叉树

//参数：后序序列和中序序列，及起始和终止位置

//返回：根节点

int countLeafSum(BiTree root)

//函数功能：计算叶子节点数

//返回：根节点

void getwidth(BiTree root,int \*count,int \*counter)

//函数功能：得到一个二叉树每层的节点数目

//参数：根节点，count为保存每层节点数目的数组，counter为保存层数的地址

int isEquivalent(BiTree bt1,BiTree bt2)

//函数功能：判断两树是否等价

//返回：等价则返回1，否则返回0

## 2.3 程序整体流程

1.整体流程，如下图1所示；

2.按先序建立二叉树，虚线用到递归（余同），如下图2所示；

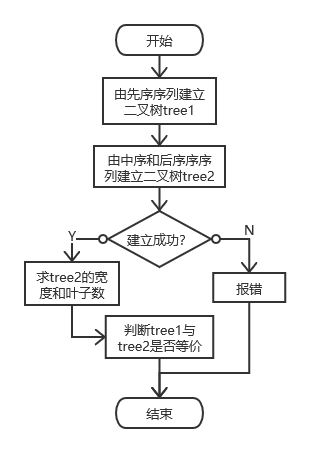
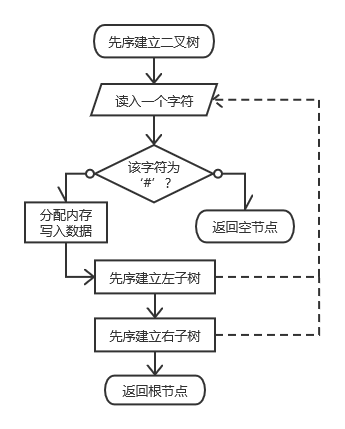
 

图 1 图 2

3.由中序和后序序列建立二叉树，如下图3；

4.求二叉树叶子节点数，如下图4；

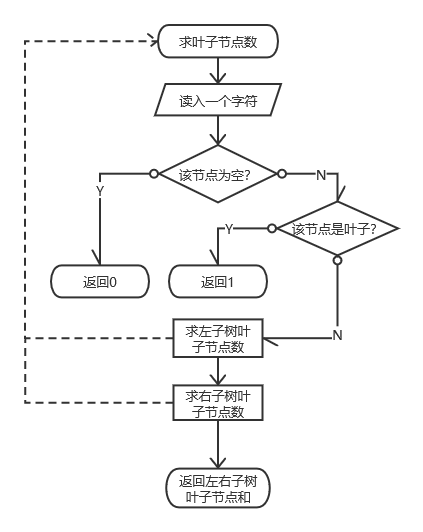
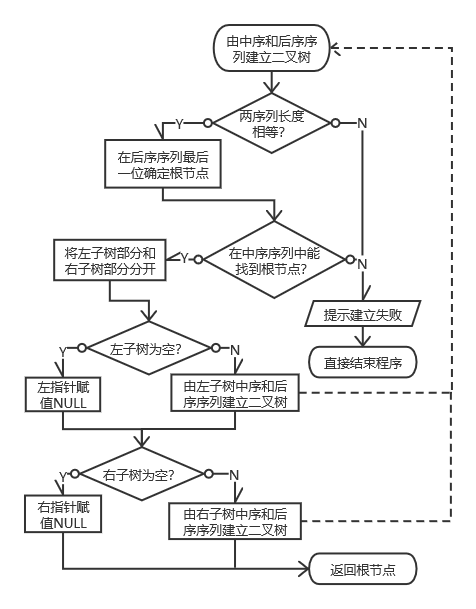


图 3 图 4

5.求二叉树宽度，如下图5；

6.判断两二叉树是否等价如下图6；

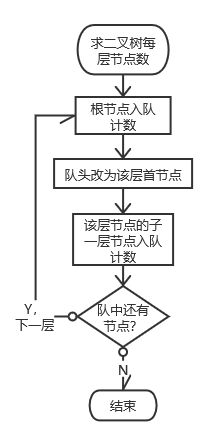
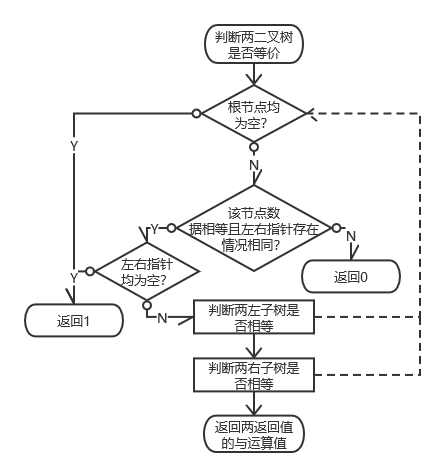
 

图 5 图 6

# 三、用户手册

欢迎使用本程序！

在本程序中，您可以进行以下操作：

（1）由前序序列建立二叉树

由提示，您可以输入二叉树前序序列建立二叉树tree1，如（其中#表示空）：

DCB##A##E##

系统会自动输出先、中、后序遍历序列方便检查是否输入正确。

（2）由中序和后序序列建立二叉树

接下来，由提示，您可以分别输入中序和后序遍历序列建立二叉树tree2，如：

DECBHIGJFA

DCEBAHGIFJ

若两序列无法建立二叉树，系统会有相关提示，并自动退出程序。

（3）计算二叉树tree2的叶子节点数和宽度

此过程无需输入，自动进行。

（4）判定二叉树tree1与tree2是否等价

此过程无需输入，自动进行结果为1则表示两二叉树等价；结果为0则表示两二叉树不等价。

感谢使用本程序，欢迎提出宝贵意见！

# 四、结果

1．测试样例一，如下图7所示；

2．测试样例二，如下图8所示；

3．测试样例三，如下图9所示。

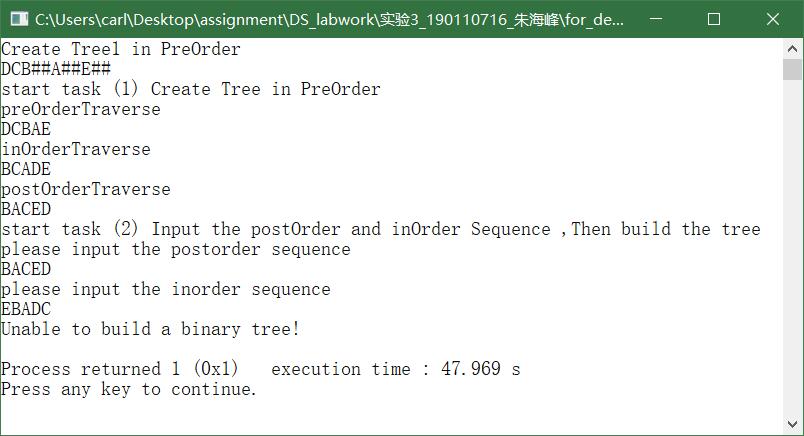


图 7

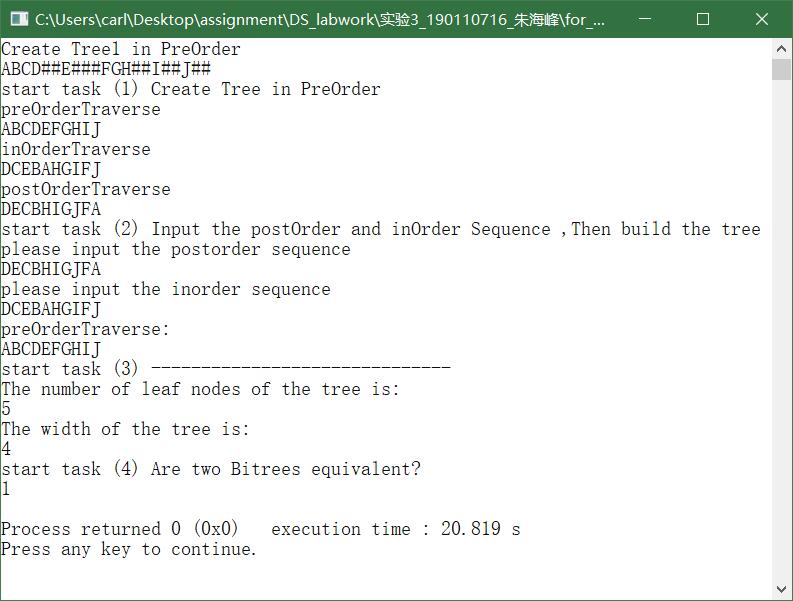


图 8

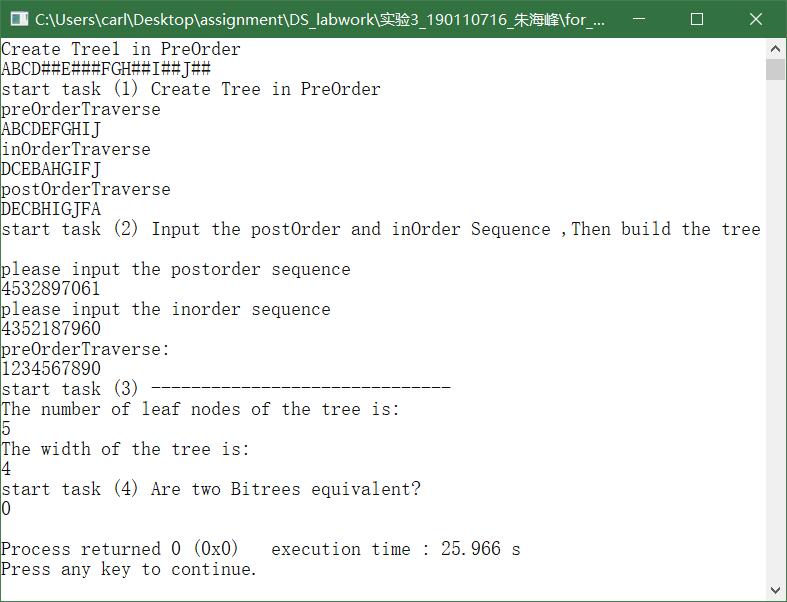


图 9

# 五、总结

该实验主要数据结构涉及到链式二叉树，和层序遍历时用到的队列。主要算法，有二叉树的遍历、由序列建立二叉树、计算二叉树叶子节点数、计算二叉树宽度和判断两个二叉树是否等价。**算法思想主要是递归。**

遇到问题无非是调试的时候，又发现以下问题：

一、内存访问错误，左右子树递归语句相似，复制之后没有及时更改造成；

二、边界情况，如到最末端的边界情况考虑空指针不够周全等小的问题；

三、函数传参遇到问题，一般可传值、传指针和引用传参，这次那个层序遍历的总层数本想引用传参直接搞定，不知为何报错，尚不清楚，只好改成用指针解决。

当然实验也有收获，其实算法都不难，就是要自己手打一遍，一呢加深印象，二呢，心里有个底——肯定会了，三呢多打代码，多练手。