**先序中序二叉树建立实验**

班级：计算机系1班 姓名：杨林卓 学号：PB15111629

题目：建立二叉树

1. **需求分析**
2. 输入二叉树先序和中序的两个字符串，根据这两个字符串建立二叉树
3. 输出二叉树的后序遍历结果
4. **概要设计**
5. 二叉树结点数据类型

class BiNode

{

public:

char ch; //存数二叉树中存放的字符

BiNode \*left, \*right; //二叉树左孩子，右孩子指针

BiNode(char c = 'a', BiNode \*l = 0, BiNode \*r = 0) //初始化

{

ch = c;

left = l;

right = r;

}

~BiNode() //销毁

{

delete left;

delete right;

left = right = 0;

}

};

typedef BiNode\* BiTree;

1. 建立二叉树的函数

int Position(char \*str, char ch); //返回ch在str中的位置

void PreMidCreate(BiTree &Node, char \*pre, char \*inorder, int length);

//根据前序中序序列建立二叉树

void PostTraverse(BiTree T); //后序遍历二叉树

int main(); //主程序

**三、函数详细设计**

（1）int Position(char \*str, char ch) //在str字符串中寻找字符ch，

{ //返回它的相对str的偏移量

int length = strlen(str);

for (int i = 0; i < length; i++)

{

if (str[i] == ch)

{

return i;

}

}

return -1; //-1表示字符串中不存在该字符

}

（2）void PreMidCreate(BiTree &Node, char \*pre, char \*inorder, int length)

{ //前序中序建立二叉树

if (length <= 0) //字符串长度为零直接返回

return;

Node = new BiNode;

Node->ch = \*pre;

int i = Position(inorder, \*pre); //寻找pre中的字符在中序序列中的位置

if (i > -1)

{

//递归生成左子树，将pre中字符在中序序列左边的字符看成新的树

PreMidCreate(Node->left, pre + 1, inorder, i);

//递归生成右子树，将pre中字符在中序序列右边的字符看成新的树

PreMidCreate(Node->right, pre + i + 1, inorder + i + 1, length - i - 1);

}

}

（3）void PostTraverse(BiTree T)

{ //后序遍历树并输出结果

if (!T)

{

return;

}

PostTraverse(T->left);

PostTraverse(T->right);

cout << T->ch << " ";

}

（4）主程序

int main()

{

char pre[50]; //储存前序序列

char inorder[50]; //储存中序序列

cout << "Enter the pre: ";

cin >> pre;

cout << "Enter the inorder:";

cin >> inorder;

BiTree root;

PreMidCreate(root, pre, inorder, strlen(inorder)); //建立二叉树

PostTraverse(root); //后序遍历输出结果

return 0;

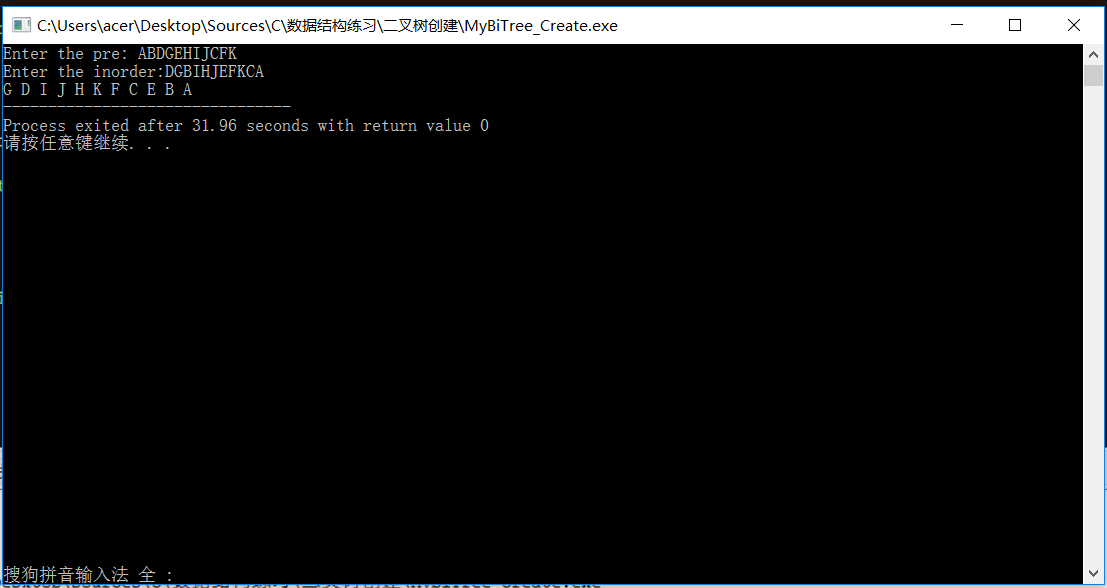
}

**四、调试分析**

调试中遇到的主要问题是对PreMidCreate函数传递参数时出错，尤其是传递字符串的起始位置以及字符串的长度时容易出现错误，导致结果错误。

因为是用递归编写的函数，所以在思维上的障碍不是很大，主要还是细心的问题。

**五、测试结果**



测试数据为：

前序：ABDGEHIJCFK

中序：DGBIHJEFKCA

输出后序结果为：GDIJHKFCEBA

测试结果成功。

**六、附录**

MyBiTree\_Create.cpp