**数据结构课程设计**

**-家谱管理系统**

# 题目描述

1. 项目简介

家谱是一种以表谱形式，记载一个以血缘关系为主体的家族世袭繁衍和重要任务事迹的特殊图书体裁。家谱是中国特有的文化遗产，是中华民族的三大文献（国史，地志，族谱）之一，属于珍贵的人文资料，对于历史学，民俗学，人口学，社会学和经济学的深入研究，均有其不可替代的独特功能。本项目兑对家谱管理进行简单的模拟，以实现查看祖先和子孙个人信息，插入家族成员，删除家族成员的功能。

1. 项目功能要求

本项目的实质是完成兑家谱成员信息的建立，查找，插入，修改，删除等功能，可以首先定义家族成员数据结构，然后将每个功能作为一个成员函数来完成对数据的操作，最后完成主函数以验证各个函数功能并得到运行结果。

# 总体思路

家谱天然的树状结构让我们第一时间想到用树来当存储结构，而一个父亲节点的儿子节点不止一个，因此此树为多叉树。相比于[二叉树](https://so.csdn.net/so/search?q=%E4%BA%8C%E5%8F%89%E6%A0%91&spm=1001.2101.3001.7020),只需要写一个结构体,在结构体中添加左节点、右节点就可以表示整颗树,但是多叉树有多个孩子结点,无法仅用左右孩子做到。对于多叉树我们常常用左子女右兄弟的二叉树来表示，一个节点的左子节点代表他的儿子，右子节点代表他的兄弟。那么我们所有的动作就是在对一颗二叉树进行操作，由此转换为对一颗二叉树增删改查的代码编写。

# 代码类设计

主文件包含了家谱系统的菜单设计和操作数执行分支。

头文件tree.h中包含了树节点结构体TreeNode的定义、对左子女右兄弟二叉树的类定义Tree和一个简单的自定义string类用于输入成员名字。

以下是类Tree的定义：



# 具体实现

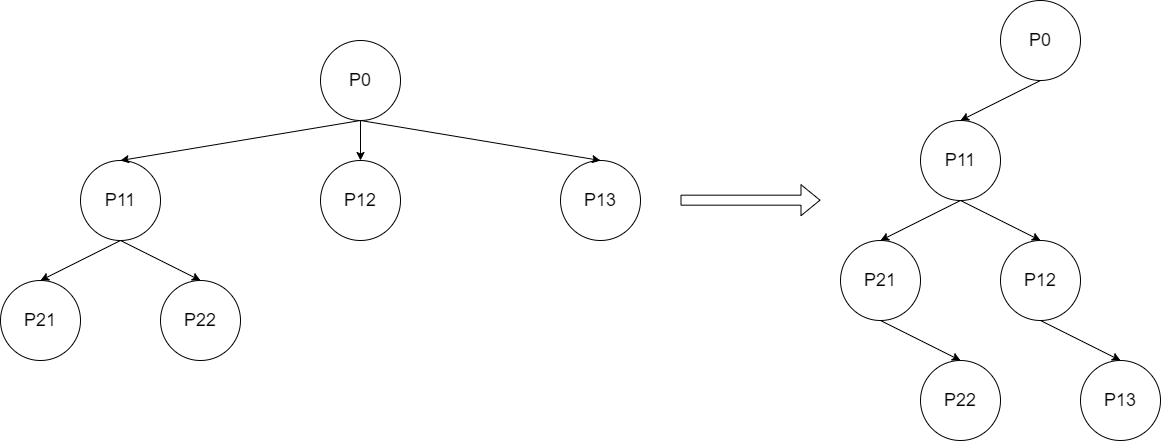
## 输入处理

根据菜单的不同操作数，在范围外的先排除，退出程序和打印二叉树单独分支。该项目的输入提示和输入反馈较多，因此其他对家谱进行增删改查的功能均包装在一个函数中一起判断。所有操作均有对现有成员的查找，所以实现功能前需要判断该成员是否存在。

本次输入需要输入家族成员的名字，因此写了一个简易的只有等于和不等于、输入和输出、赋值等号的运算符重载。

## 左子女右兄弟二叉树的构建

树节点结构体TreeNode为三元结构体，包含了该节点成员的姓名，左子节点firstChild和右子节点nextSibling，分别代表父节点的子女和兄弟。在Tree类中，有root指针作为树的根节点，由此建树。根据以上原理，左子女右兄弟二叉树与多叉树的转换如下。



### 查找成员

在本项目中多次有查找某个成员节点的情况，比如组建家庭、添加新成员和解散家庭，都需要用户输入父节点信息。而在树中，遍历会有两种，即深度优先遍历和广度优先遍历。由于左子女右兄弟二叉树的结构偏左，我么选择深度优先遍历中的先根次序遍历。思想是从根节点出发，依次先根次序遍历根的各个子树。查找代码如下：

在Tree的类设计中有指针current表示树当前所浏览的节点，这有便于寻找当前节点的左子女和右兄弟。比如如果在添加家庭成员的操作中，在不设current指针的情形下，每次都必须从头开始遍历逐个节点比较，时间复杂度会很高；而设立了current的情况下，每次新插入子女可以不用从头开始检测，从current所指位置开始寻找，可以减少很多次比较。

有时候查找未必需要用到此节点，只是为了确定该节点是否存在，比如添加成员与现有成员不重名，因此需要给定参数决定是否将找到的节点赋值为当前节点。

### 添加成员

判断添加的成员在家谱中是否存在，不接受不同人相同姓名。先用上述搜索方式查找到添加成员的位置。添加新成员往往有两种位置，一个是作为添加位置的左子女，第二是作为添加位置的右兄弟，因此设立了两个函数和 分别实现。其中添加右兄弟的思路为找到当前插入位置右分支的最末端，添加该成员到它的右节点。而添加左子女有两种情况，一是父节点的第一个儿女，二是父节点的第二胎或以后，前者直接令父节点左子女指针指向新成员即可，后者调用添加右兄弟的函数，第一个参数设定为父节点的第一个左子女即可。

### 删除成员的左子树

由题意可知解散一个成员的家庭，我们记他为P，指的是将P所有后代从家谱树中删除，而自身不删除。如此我们只需要对以P作为根节点的子树做后序遍历，如此可以在子节点的指针都释放了再释放根节点指针，防止左指针释放后找不到右指针去释放而生成野指针，如此可以保证所有指针被正确释放，最后还要对P的左子女指针赋值为空。



### 修改成员

只需要找到名字为曾用名的节点位置，修改为现用名即可。主体就是第一部分的搜索。

### 打印家谱

一个祖先会有多个儿女，每个儿女又会各自组建家庭，输出树结构时需要将辈分高的放在其后代前。考虑输出自己后，要先输出自己的儿女和所有后代，再输出自己兄弟的家庭情况，这天然的符合对左子女右兄弟二叉树的前序遍历。前序遍历正是先输出根节点，再输出左子树即自己的所有后代，再输出右子树即自己兄弟的家庭。前序遍历的函数放在私有成员函数中，在公开函数部分添加调用它的函数。

打印家谱的代码如下：

# 项目小结

1. 性能分析

树节点定义有三个变量，一共n个节点，总共空间复杂度为。在寻找方面，如果要访问节点x的第k个子女，只需要从开始沿着链继续走k-1步即可完成，时间复杂度为，d为树的度。若要寻找父节点，就必须遍历树，时间复杂度为。

若要在某个子女链中按照有序的次序不断插入节点，由于current指针的设置，可以不必从根节点逐个查找，不会使得时间复杂度达到，而减少到常数级。

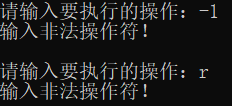
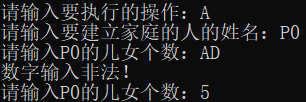
1. 优化

在解散家庭时我们的操作是对子树进行后序遍历；打印家谱树时我们的操作是前序遍历整棵树。这两种遍历都是递归形式的，而树的递归遍历可以优化为非递归形式。如此可以减少递归栈的使用，提高效率。而对树进行深度优先遍历时用的也是递归函数，也可对此进行非递归转化。

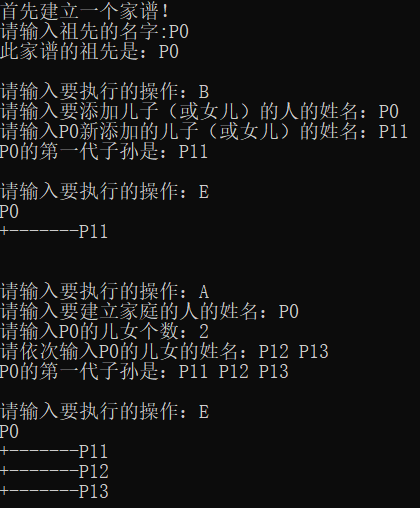
当然对于左子女右兄弟链表示法可以用带双标记的先根次序表示法存储结构，这样可以简化或省略专门用来表明节点之间关系的指针，使得内存进一步减少。但由于家谱树需要添加新成员和解散家庭的需求，不只是查找遍历，所以此方法不适合此题。

# 程序功能测试

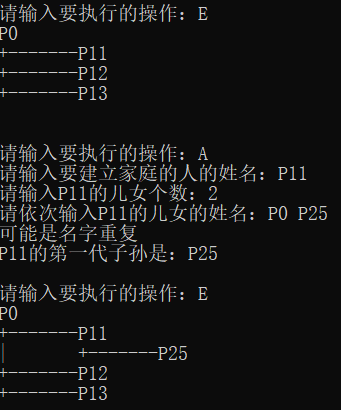
1. 错误输入

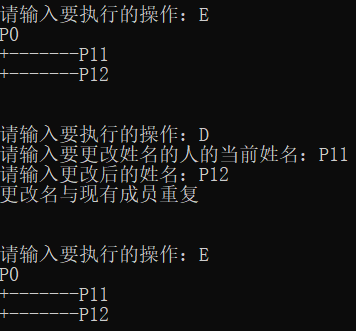
1. 意外情况
2. 添加子女在完善家庭前进行，均默认是追加子女



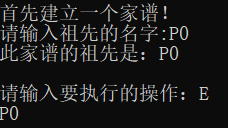
1. 添加子女有重名的，跳过重名的继续添加



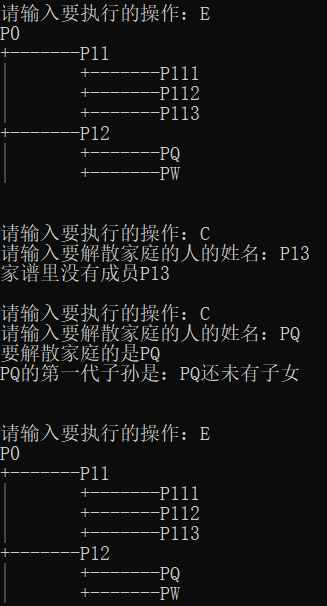
1. 更改名字与现有成员重名



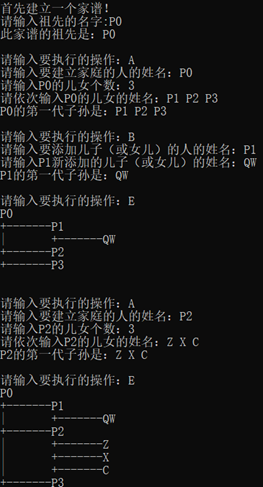
1. 打印空树



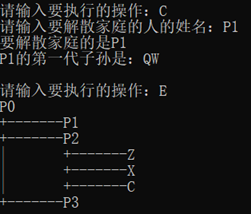
1. 解散未知家庭和无子女的家庭



1. 正常情况
2. 添加 成员，两种方式



1. 解散家庭



1. 更改姓名

