项目说明文档

**数据结构课程设计**

**——家谱管理系统**

作 者 姓 名： 吴英豪

学 号： 1953608

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji University

1.分析

1.1背景分析

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人物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。

1.2功能分析

本项目的实质是完成对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。

同时还具有简单的输出人物关系的功能。对于族谱中的某一个成员，可以查询他的父母(父结点)，兄弟姐妹(兄弟结点),子女(子女结点)的相关信息。

2.设计

2.1数据结构设计

此系统主要考察树这种数据结构。考虑到一个家族中一个家庭成员和另一个家庭成员的关系可能是祖先、父母、兄弟、子女、后代等等。所以笔者采用“左长子，右兄弟”的森林结构。从根结点开始，对于森林中的某一个结点，其左子树上的结点均为此结点的子女；其右子树上的结点均为此结点的兄弟(或姐妹)。

出于优化系统的目的，笔者采用了链表栈的数据结构，栈主要用于用户输入时暂存输入的成员姓名；在用户输入结束之后栈将弹出元素，向用户展示自己刚才输入的成员姓名，方便用户和系统进行交互。

2.2类结构设计

引入类FamilyTree来表示森林这种数据结构,引入树节点FamilyNode类，作为森林的结点,引入Stack类，表示栈；将Stack设计为链表栈的形式，方便元素的入栈与出栈，Stack类的结点为FamilyNode.

考虑到增强代码的复用性，所有的类均采用模板的形式，FamilyNode<T>,

Stack<T>,FamilyTree<T>.

由于栈Stack和森林FamilyTree需要访问其结点内部的数据，所以将Stack和FamilyTree设置为FamilyNode的友元

2.3成员与操作设计

(1)结点(FamilyNode)

//家谱结点

template<class T>

class FamilyNode

{

private:

T \_name; //结点的名称

FamilyNode<T>\* \_childNode; //左节点-子女结点

FamilyNode<T>\* \_siblingNode; //右节点-兄弟结点

FamilyNode<T>\* \_preSiblingNode; //前兄弟结点

FamilyNode<T>\* \_next; //新增，栈中的下一个元素

FamilyNode<T>\* \_parentNode; //新增，指向父节点

public:

FamilyNode();

FamilyNode(T data);

~FamilyNode();

T getNodeName() const;

friend FamilyTree<T>;

friend Stack<T>;

};

(2)森林(FamilyTree)

//家谱

template<class T>

class FamilyTree

{

private:

FamilyNode<T>\* \_first; /\*家谱的内容一律存放在\_first的\_childNode指针域中，\_silbNode始终为空\*/

protected:

void recursionDelete(FamilyNode<T>\*node); //递归删除结点

public:

FamilyTree();

~FamilyTree();

FamilyNode<T>\* getNodeByName(T name,FamilyNode<T>\* node,bool& tag) ; //查询名为name的结点

FamilyNode<T>\* getFirstNode()const //返回祖先结点（第一个有用的结点）

{

return this->\_first->\_childNode;

};

void insertNode(T target, T name, Tag tag); //在名为target的结点处插入名为name的tag类型结点

void changeNode(T target,T name); //将名为target的结点名称修改为name

void insertRoot(T name); //插入根节点

void printNode(FamilyNode<T>\* number); //输出结点number的兄弟和子女

void deleteNode(FamilyNode<T>\* node);

};

(3)栈(Stack)

//栈 以树结点为栈结点

template<class T>

class Stack

{

private:

FamilyNode<T>\* \_top;

public:

Stack();

~Stack();

void push(T name); //用于类外入栈

void pop(); //用于.cpp的函数

friend class FamilyTree<T>;

};

2.4文件设计

本系统支持windows系统和Linux系统。

(1)Windows系统

windows系统的相关文件在文件夹06\_1953608\_WuYinghao\_Windows中，

其中类的定义以及其成员函数的定义在06\_1953608\_WuYinghao.h中;main函数以及相关的函数定义在06\_1953608\_WuYinghao.cpp中.06\_1953608\_WuYinghao.exe为Windows平台下的可执行文件。

**06\_1953608\_WuYinghao.cpp和06\_1953608\_WuYinghao.h的编码格式均为**

**简体中文(GB2312)-代码页 936**

(2)Linux 系统

Linux系统的相关文件在文件夹06\_1953608\_WuYinghao\_Linux中,

其中类的定义以及其成员函数的定义在06\_1953608\_WuYinghao\_Linux.h中;main函数以及相关的函数定义在06\_1953608\_WuYinghao\_Linux.cpp中.06\_1953608\_WuYinghao.out为Linux平台下的可执行文件。

**06\_1953608\_WuYinghao\_Linux.cpp和06\_1953608\_WuYinghao\_Linux.h的编码格式均为Unicode(UTF-8 无签名)-代码页65001**

3.实现

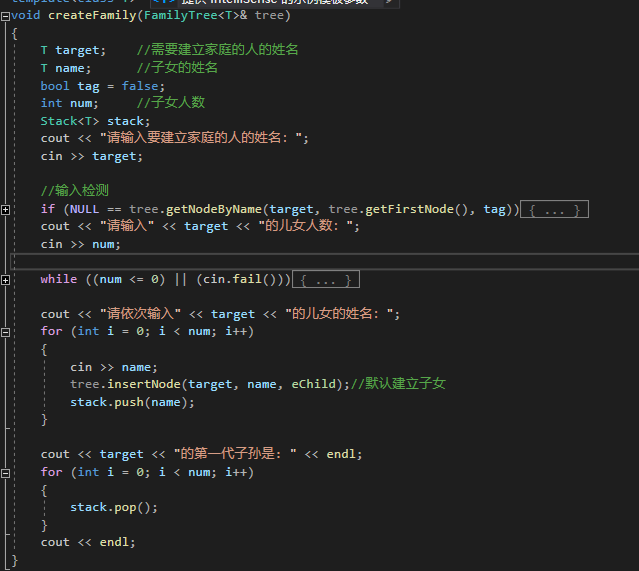
3.1 完善家谱功能(A)测试

3.1.1 功能说明

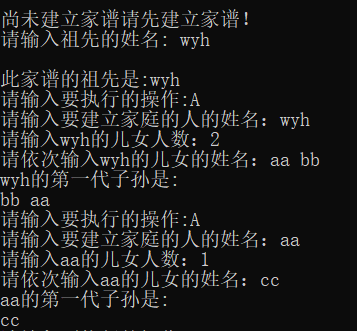
此功能主要用于系统的初始化，刚进入系统时首先需要用户输入祖先的相关信息，然后才能进行后续功能的实现。

除了初始化以外，在后续阶段此操作可以实现给结点添加子女的功能(只能添加子女，不能添加兄弟)

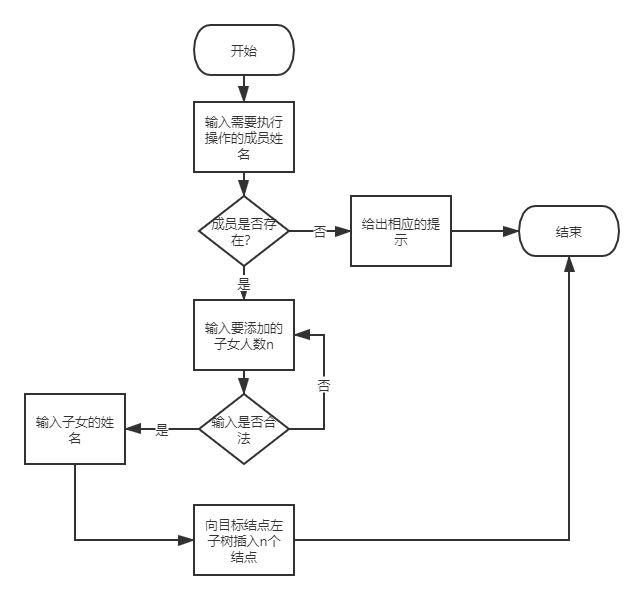
3.1.2 核心代码展示



3.1.3 程序运行截图展示



3.1.4 流程图展示

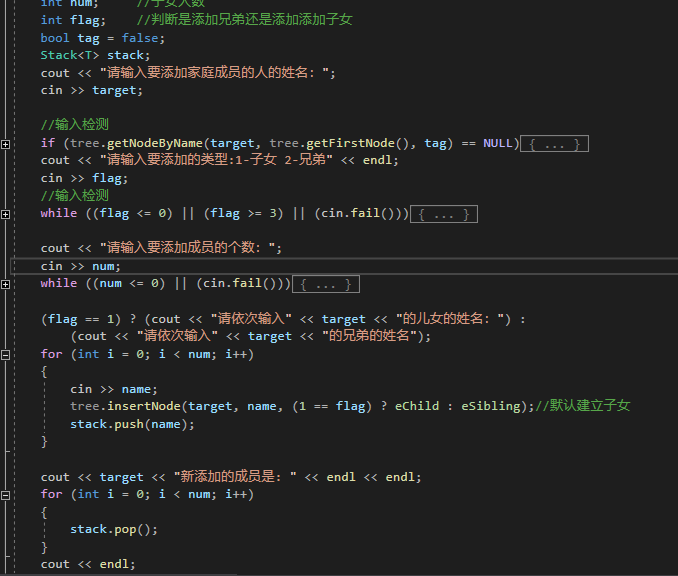


3.2 添加家庭成员功能(B)测试

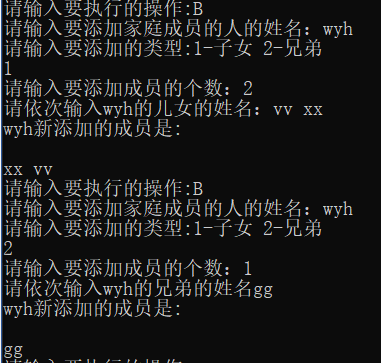
3.2.1 功能说明

此功能能够让用户给结点添加子女或兄弟

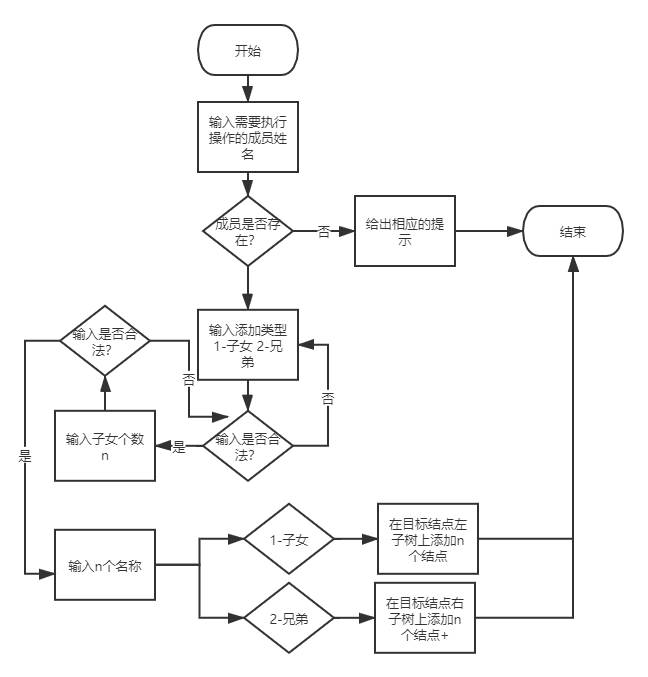
3.2.2 核心代码展示



3.2.3 程序运行截图展示



3.2.4 流程图展示

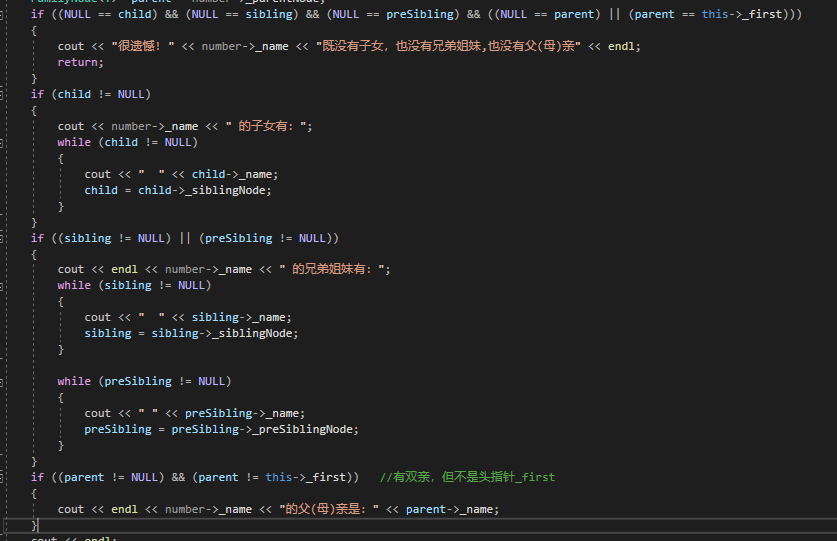


3.3 查看家庭成员功能(C)测试

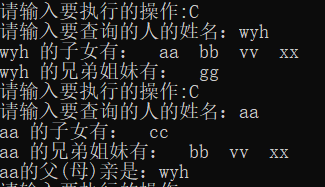
3.3.1 功能说明

此功能能够让用户查看某一个家谱成员的相关信息(子女、父母、兄弟的姓名)。查看时只需要输入成员的姓名即可

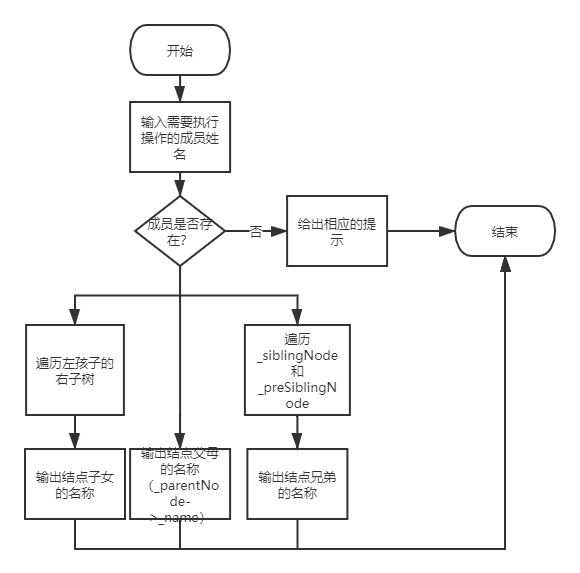
3.3.2 核心代码展示



3.3.3 程序运行截图展示



3.3.4 流程图展示



3.4 解散局部家庭功能(D)测试

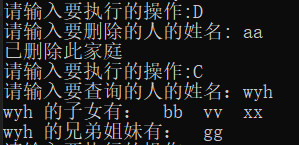
3.4.1 功能说明

此功能能够让用户删除家谱中的某个结点，删除结点时，将把这个结点以及其左子树(子女分支)的所有结点删去，其右节点(兄弟分支)将会被保留。

3.4.2 核心代码展示

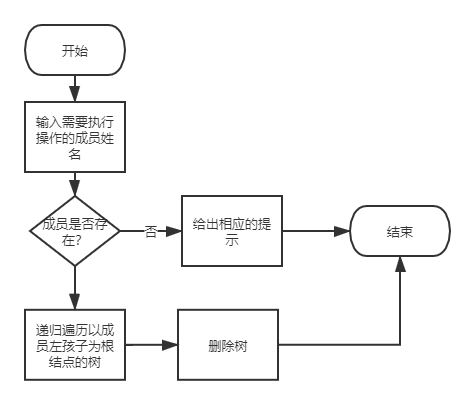


3.4.3 程序运行截图展示



可以发现删除aa结点后其左子树被完全删除，其右子树(兄弟分支bb等)被保留

3.4.4 流程图展示

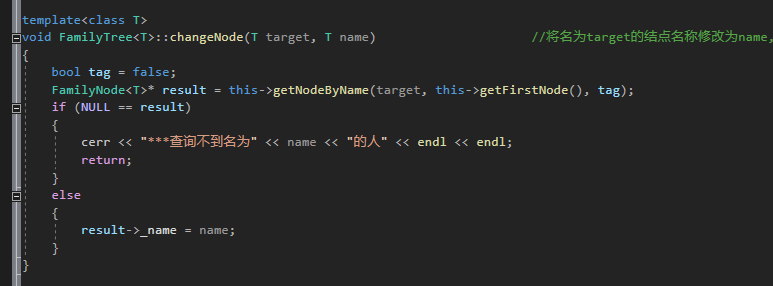


3.5 更改家庭成员姓名功能(E)测试

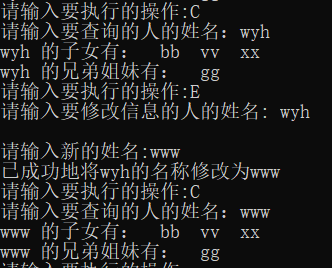
3.5.1 功能说明

此功能支持用户修改家谱中某个成员的姓名，首先输入原来的姓名，在提示下输入新的名称就可以实现名称更改。

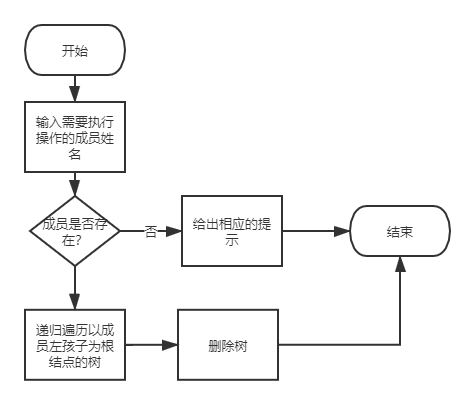
3.5.2 核心代码展示



3.5.3 程序运行截图展示

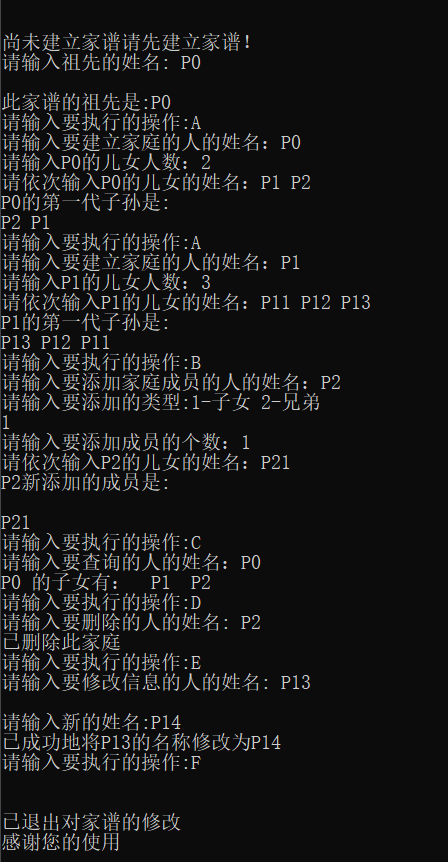


3.5.4 流程图展示



**4.测试**

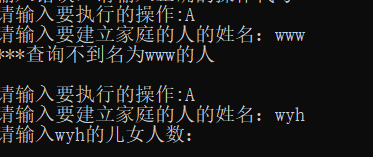
4.1 样例测试



4.2 边界测试与错误测试

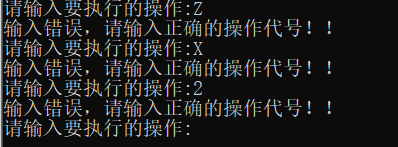
4.2.1 查询不到家谱中的人

执行删除、查询、添加子女等时需要输入家谱成员的姓名，若输入一个不存在于家谱中的名字，系统会给出相应的提示。



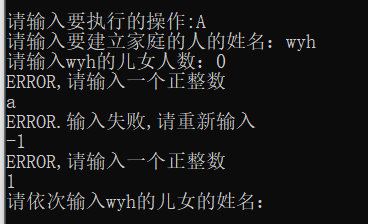
4.2.2输入不合法的操作指令

此系统所支持的操作指令为A、B、C、D、E、F 若输入不合法的指令，系统能给出相应的提示。



4.2.3 输入正整数时输入不合法的值

此系统的插入子女等操作设计输入人数（正整数）,若输入非法的值，系统会给出相应的提示。



4.2.4 执行B操作时输入非法值

此系统的B操作(添加家庭成员)，需要用户输入需要添加的家庭成员类型

(1-子女2-兄弟)。若输入非法值，系统会给出相应的提示。

