家谱管理系统

问题描述

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。

项目要求

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。

功能实现及代码分析

1、类设计

用二叉树实现多叉树家谱统计，结点保存姓名，左右子节点分别是第一个儿女和下一个同辈。

Tree类包含家谱系统的各功能，如建立家庭、完善家庭、解散局部家庭、添加家庭成员、更改成员信息等。

```
class person //定义节点类
{
    friend class Tree; //家族树类为友元
public:
    person() :name("?"), firstChild(NULL), nextSibling(NULL){};
private:
    string name;
    person* firstChild;
    person* nextSibling;
};

class Tree
{
public:
    Tree() :root(NULL){};   //构造函数
    void create(Tree& L);   //建立家庭
    void complete(Tree& L); //完善家庭
    void dismiss(Tree& L);  //解散家庭
    void kill(person* p);   //删除节点
    void insert(Tree& L);   //添加家庭成员
    void update(Tree& L);   //更改成员信息
    void print(person* p);  //输出第一代子孙
    person* search(person* p, string name); // 在家谱中搜索成员

    person* root;
};
```

2、进入项目

进入项目，引导用户输入祖先新建家谱，并可多次选择要执行的操作，直到输入E退出程序。

```
int main() {
    cout << "**          家谱管理系统           **" << endl;
    cout << "================================" << endl;
    cout << "**       请选择要执行的操作:       **" << endl;
    cout << "**         A.完善家谱              **" << endl;
    cout << "**         B.添加家庭成员          **" << endl;
    cout << "**         C.解散局部家庭          **" << endl;
    cout << "**         D.更改家庭成员姓名       **" << endl;
    cout << "**         E.退出程序              **" << endl;
    cout << "================================" << endl;


    cout << "首先建立一个家谱!" << endl;
```

```
    Tree family;
    family.create(family);
    char mark = ' ';
    while(mark != 'E') {
        cout << "\n请选择要执行的操作:";
        cin >> mark;
        switch(mark) {
            case'A': {  //完善家谱
                family.complete(family);
                break;
            }
            case'B': {  //添加家庭成员
                family.insert(family);
                break;
            }
            case'C': {  //解散局部家庭
                family.dismiss(family);
                break;
            }
            case'D': {  //更改家庭成员姓名
                family.update(family);
                break;
            }
            case'E':    //退出程序
                break;
            default:
                cout << "请输入正确操作!" << endl;
        }
    }
}
```

3、创建家庭

初次创建家庭，新建节点并输出信息。

```
void Tree::create(Tree& L) { //建立家谱
    cout << "请输入祖先的名字:";
    string rootname;
    cin >> rootname;
    person* p = new person;
    p->name = rootname;
    L.root = p;               //添加为根节点
    cout << "此家谱的祖先:" << p->name << endl;
}
```

4、完善家谱

对家庭中成员初始化子孙信息。先检索该成员是否在家谱中，若不存在则报错并要求用户重新输入；若存在且有多个儿女，第一个儿女由该成员的firstChild指针指向，其余儿女结点由firstChild的nextSibling依次指向。

```
void Tree::complete(Tree& L) {
    cout << "请输入要创立家庭的人的姓名:";
    string rootname;
    cin >> rootname;
    person* s = Tree::search(L.root,rootname);
    if(s) {
        person* r = s;
        cout << "请输入" << s->name << "的儿女数:";
        int n;
        cin >> n;
        int m = n;
        cout << "请依次输入" << s->name << "的儿女的姓名:";
        while(m) {
            string na;
            cin >> na;
            person* p = new person;
            p->name = na;
            if(m == n) {
                s->firstChild = p;
                s = s->firstChild;
            } else {
                s->nextSibling = p;
                s = s->nextSibling;
```

```
            }
            m--;
        }
        Tree::print(r);
    } else {          //若要创立家庭的人不在家谱中
        cout << "查无此人，请重新输入!" << endl;
        Tree::complete(L);
    }
}
```

5、添加家庭成员

对已建立家庭的成员添加家庭成员。先检索要添加家庭成员的成员是否在家谱中，若不存在则报错并要求用户重新输入；若存在，判断firstChild指针是否为空，若为空则可以直接将新儿女信息添加，若不为空，遍历该成员儿女信息，将新儿女添加到nextSibling为空的儿女的指针处。

```
void Tree::insert(Tree& L) {
    cout << "请输入要添加子女的人的姓名:";
    string rootname;
    cin >> rootname;
    person* s = Tree::search(L.root,rootname);
    if(s) {  //添加新结点
        person* r = s;
        cout << "请输入" << s->name << "新添加的子女的姓名:";
        person* p = new person;
        string name;
        cin >> name;
        p->name = name;
        if(!s->firstChild) { //若当前节点无孩子节点
            s->firstChild = p;
        } else {
            s = s->firstChild;
            while(s->nextSibling) {
                s = s->nextSibling;
            }
            s->nextSibling = p;
        }
        Tree::print(r);
    } else {
        cout << "查无此人，请重新输入!" << endl;
        Tree::insert(L);
    }
}
```

6、解散局部家庭

函数	返回值	描述
dismiss()	void	接收用户输入，在家谱中搜索成员，若成功检索，将成员指针传给kill函数进行结点删除。
kill(person *subTree)	void	遍历删除结点。

```
void Tree::dismiss(Tree& L) {      //解散家庭
    cout << "请输入要解散家庭的人的姓名:";
    string rootname;
    cin >> rootname;

    person * subTree = search(L.root, rootname);
    if(subTree) {
        cout << "要解散家庭的人是: " << rootname << endl;
        Tree::print(subTree);
    }
    L.kill(L.search(L.root, rootname));
}

void Tree::kill(person *subTree)  //删除结点
{
    if(subTree) {                 //当前指针不为空 说明有子树
        kill(subTree->firstChild);//则递归下去
        delete subTree;           //最后删除
    }
}
```

7、更改家庭成员姓名

根据用户输入修改成员姓名。先检索家谱中是否有此人，若无则报错并要求用户重新输入；若检索到该成员，对姓名进行修改并输出信息。

```
void Tree::update(Tree& L)
{
    cout << "请输入要更改姓名的人的目前姓名:";
    string rootname;
    cin >> rootname;
    person* s = Tree::search(L.root,rootname);
    if(s) {
        cout << "请输入更改后的名字:";
        string name;
        cin >> name;
        s->name = name;
        cout << rootname << "已更改为" << s->name << endl;
    } else {  //若要求更改的成员不在家谱中，重新输入
        cout << "查无此人，请重新操作!" << endl;
        Tree::update(L);
    }
}
```

8、辅助函数

搜索成员

遍历二叉树，搜索结点信息。

```
person* Tree::search(person* p,string name) {  //在家谱中检索成员
    person* t = NULL;
    person* s[100];  //辅助数组
    int top = 0;
    while(p || top > 0) {
        while(p) {
            if(p->name == name) {
                t = p;
            }
            s[++top] = p;
            p = p->firstChild;
        }
        p = s[top--];
        p = p->nextSibling;
    }
    return t;
}
```

输出成员

规范输出工具，遍历输出第一代儿女。

```
void Tree::print(person* p) {
    cout << p->name << "的第一代子孙是:" << p->firstChild->name << "\t\t";
    p = p->firstChild;
    while(p->nextSibling) {
        cout << p->nextSibling->name << '\t';
        p = p->nextSibling;
    }
    cout << endl;
}
```

用例演示

进入项目

```
**          家谱管理系统          **
================================
```

```
**       请选择要执行的操作:        **
**          A.完善家谱              **
**          B.添加家庭成员          **
**          C.解散局部家庭          **
**          D.更改家庭成员姓名       **
**          E.退出程序              **
==================================
首先建立一个家谱!
请输入祖先的名字:P0
此家谱的祖先:P0
```

完善家谱

若输入创建家庭成员姓名不在家谱中，要求重新输入。下述实现检查功能类似，不加以赘述。

```
请选择要执行的操作:A
请输入要创立家庭的人的姓名:P0
请输入P0的儿女数:2
请依次输入P0的儿女的姓名:P1 P2
P0的第一代子孙是:P1        P2

请选择要执行的操作:A
请输入要创立家庭的人的姓名:P3
查无此人，请重新输入!
请输入要创立家庭的人的姓名:P1
请输入P1的儿女数:3
请依次输入P1的儿女的姓名:P11 P12 P13
P1的第一代子孙是:P11       P12 P13
```

添加家庭成员

```
请选择要执行的操作:B
请输入要添加子女的人的姓名:P4
查无此人，请重新输入!
请输入要添加子女的人的姓名:P2
请输入P2新添加的子女的姓名:P21
P2的第一代子孙是:P21
```

解散局部家庭

```
请选择要执行的操作:C
请输入要解散家庭的人的姓名:PPP
查无此人，请重新输入!
请输入要解散家庭的人的姓名:P2
要解散家庭的人是: P2
P2的第一代子孙是:P21
```

更改家庭成员姓名

```
请选择要执行的操作:D
请输入要更改姓名的人的目前姓名:PPP
查无此人，请重新操作!
请输入要更改姓名的人的目前姓名:P13
请输入更改后的名字:P14
P13已更改为P14
```

退出程序

```
请选择要执行的操作:E

Process finished with exit code 0
```