

**课 程 设 计**

|  |  |
| --- | --- |
| **课程设计名称：** | **数据结构课程设计** |
| **专 业 班 级 ：** | **计算机科学与技术2021级2101班** |
| **学 生 姓 名 ：** | **华明洋** |
| **学 号 ：** | **211040100109** |
| **指 导 教 师 ：** | **赵晨阳** |
| **课程设计时间：** | **2022.12.5-2022.12.16** |

**计算机科学与技术 专业课程设计任务书**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **学生姓名** | **华明洋** | **专业班级** | | **计科2101** | **学号** | | **211040100109** |
| **题 目** | **家谱管理系统** | | | | | | |
| **课题性质** | **工程设计** | | **课题来源** | | | **自拟课题** | |
| **指导教师** | **赵晨阳** | | **同组姓名** | | | **无** | |
| **主要内容** | **基本功能:**  按需求完成基本的增、删、查、改、统计、排序等系统管理操作。   1. 保存功能：建立家族关系并能存储到文件中； 2. 增加删除功能：实现家族成员的添加、删除；   （3）查询功能：可以查询家族成员的双亲、祖先、兄弟、孩子和后代等信息。  （4）显示功能：按某种顺序输出家谱信息、输出家谱资料。  **扩展功能：**  （1）从文件中读取树，得到文件中的家谱信息  （2）家谱数据以文件格式存储。  （3）能用括号表示法输出家谱二叉树。 | | | | | | |
| **任务要求** | （1）能够根据需求，选择合适的数据结构，确定系统的设计目标。  （2）结合软件工程的系统开发流程，利用相关开发工具，完成系统概要设计和详细设计。  （3）结合选题利用数据结构相关知识，使用C语言或C++实现该系统的所有上述功能。要求界面友善，程序运行正常。  （4）能够在设计过程中撰写规范的设计报告，在设计完成后通过答辩，并提供可运行的系统源代码电子版一套。 | | | | | | |
| **参考文献** | [1] 严蔚敏.《数据结构（C语言版）》[M]. 北京：清华大学出版社, 2018.  [2] 谭浩强.《C语言程序设计》[M]. 北京：清华大学出版社, 2018  [3]徐慧民等.C++大学基础教程[M].北京：人民邮电出版社，2005.  [4]王红梅，胡明，王涛.数据结构（C++版）.北京：清华大学出版社，2011.  [5] 苏仕华.数据结构课程设计. 机械工业出版社 | | | | | | |
| **审查意见** | **指导教师签字：**  **教研室主任签字：**  **2022年 12 月 2 日** | | | | | | |

**目录**

[1 需求分析 1](#_Toc28931)

[1.1 系统概述 1](#_Toc26485)

[1.2 系统中的角色 1](#_Toc29213)

[1.3 功能性需求 1](#_Toc20121)

[1.4 使用例图 2](#_Toc12411)

[2 概要设计 3](#_Toc7876)

[2.1 抽象数据类型 3](#_Toc27211)

[2.2 存储结构设计 4](#_Toc3580)

[2.3 功能结构图 4](#_Toc29641)

[2.4 系统子程序及功能设计 5](#_Toc11322)

[3 运行环境 6](#_Toc17223)

[4 开发工具和编程语言 6](#_Toc16359)

[5 详细设计 6](#_Toc9376)

[5.1 添加家谱模块 7](#_Toc27211)

[5.2 删除家谱模块 9](#_Toc3580)

[5.3 查询家谱模块 9](#_Toc29641)

[5.4 显示家谱模块 1](#_Toc11322)1

[5.5 保存家谱模块 1](#_Toc27211)2

[6 运行实现 1](#_Toc9600)2

[7 心得体会 1](#_Toc30198)5

[8 参考文献 1](#_Toc27534)6

**1 需求分析**

目前，很多家庭家谱丢失，家庭一些关系混乱，为了避免这一问题，制定一家谱程序。家谱用于记录某家族历代家族成员的情况与关系，实现对一个家族所有的资料进行收集整理。支持对家谱的存储、更新、查询、统计等操作。并用计算机永久储存家族数据，方便随时调用。

* 1. **系统概述**

本系统实现家谱的构建和查询，面向的对象是家族成员。对于族人管理员而言，系统对其提供信息查询和数据的及时更新工作以实现对系统的定期维护，可以针对家谱信息、成员信息进行添加、删除、查询等操作，保证系统运行的稳定性；对于族人而言，系统对其提供信息查询功能，他们可以查询所有祖先的信息和所有儿子的信息。

本系统实现对家谱信息的管理：

（1）建立家族关系并能存储到文件中；

（2）实现家族成员的添加、删除功能；

（3）可以查询家族成员的双亲、祖先、兄弟、孩子和后代等信息。

（4）按某种顺序输出家谱信息（树的遍历操作）、以树型结构输出家谱资料等功能。

* 1. **系统中的角色**

|  |  |  |
| --- | --- | --- |
| 角色 | 类别 | 描述 |
| 族人 | 服务对象 | 查询家族成员信息 |
| 家谱管理员 | 管理人员 | 增加、删除、查询、显示家族成员信息 |

* 1. **功能性需求**

**1.3.1 用户类型**

1. 家谱管理员：可在家谱管理系统上进行增加、删除、查询、显示家庭成员的操作。
2. 族人：可在家谱管理系统上进行长辈查询的相关操作。

**1.3.2 增加信息**

用户一开始有两种选择，一是创建新家谱，家谱管理员通过输入父亲母亲孩子的名字，要求在输入家谱记录时按从祖先到子孙的顺序输入。并且第一个家谱记录的父亲域为所有人的祖先。二是选择文件存储的家谱进行操作。

**1.3.3 删除信息**

用户通过输入编号操作来将系统中的家谱进行删除。选择清除记录后，程序会对应出现清除文件完成。

**1.3.4 查询信息**

用户选择查询功能，通过输入相应的编号来查询某人的所有儿子或查询某人的所有祖先。在输入族人名称时，应保证名称的存在性，当所输入名称不存在时，显示查无此人。

**1.3.5 显示信息**

用户通过选择编号展示家谱信息功能，通过括号表示法输出含有所有家庭成员的家谱信息信息。

* 1. **使用例图**

如图1：

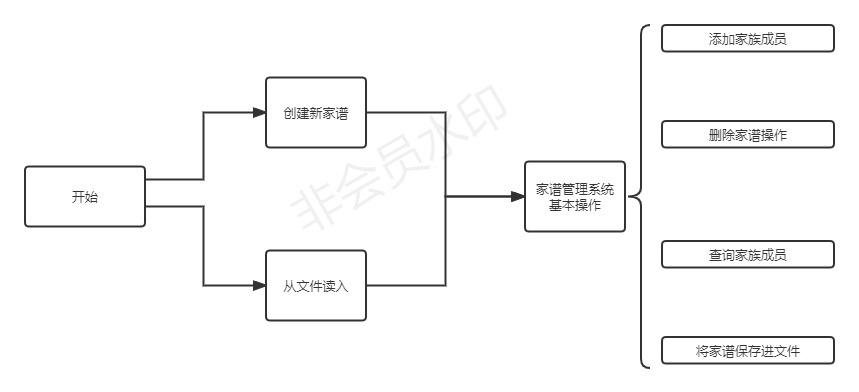


图1-1

**2 概要设计**

由于家谱是一颗树结构，而不是一颗二叉树，所以在存储时要转换成二叉树的形式，规定一个父亲结点的左孩子结点表示母亲结点(父亲结点无右孩子结点)，母亲结点的右子树表示他们的所有儿子。

**2.1 抽象数据类型**

ADT BinaryTree{

数据对象D：D是具有相同特性的数据元素的集合。

数据关系R：

若D=Φ，则R=Φ，称BinaryTree为空二叉树；

若D≠Φ，则R={H}，H是如下二元关系；

（1）在D中存在惟一的称为根的数据元素root，它在关系H下无前驱；

（2）若D-{root}≠Φ，则存在D-{root}={D1,Dr}，且D1∩Dr =Φ；

（3）若D1≠Φ，则D1中存在惟一的元素x1，<root,x1>∈H，且存在D1上的关系H1 ⊆H；若Dr≠Φ，则Dr中存在惟一的元素xr，<root,xr>∈H，且存在上的关系Hr ⊆H；H={<root,x1>,<root,xr>,H1,Hr}；

（4）(D1,{H1})是一棵符合本定义的二叉树，称为根的左子树；(Dr,{Hr})是一棵符合本定义的二叉树，称为根的右子树。

基本操作 P：

void ShowSon(TreeNode \*node,string name)；

TreeNode\* Find(TreeNode \*node,string name)；

void save(TreeNode \*root)；

void showFamily(TreeNode \*node)；

void freeTree(TreeNode \*node)；

void saveInFile(TreeNode \*node)；

void printBinaryTree(TreeNode \*node)；

}ADT BinaryTree

**2.2 存储结构设计**

本系统定义3个结构体数据，分别存储家庭成员名字，左孩子和右孩子。定义人名的元素类型，和声明了两个struct TreeNode类型的指针，分别叫lChild和rChild。

struct TreeNode{

string name;

struct TreeNode \*lChild;

struct TreeNode \*rChild;

};

**2.3 功能结构图**

根据需求分析，为了满足用户的功能需求，按照模块划分原则，系统用户分为两大类，家谱管理员和家庭成员，管理员界面可以实现添加家庭成员、删除家庭成员等功能；家庭成员可实现查询祖先信息、查询后代信息、通过括号表示法显示所有家庭成员等功能。各模块之间的关系如图2-1所示。

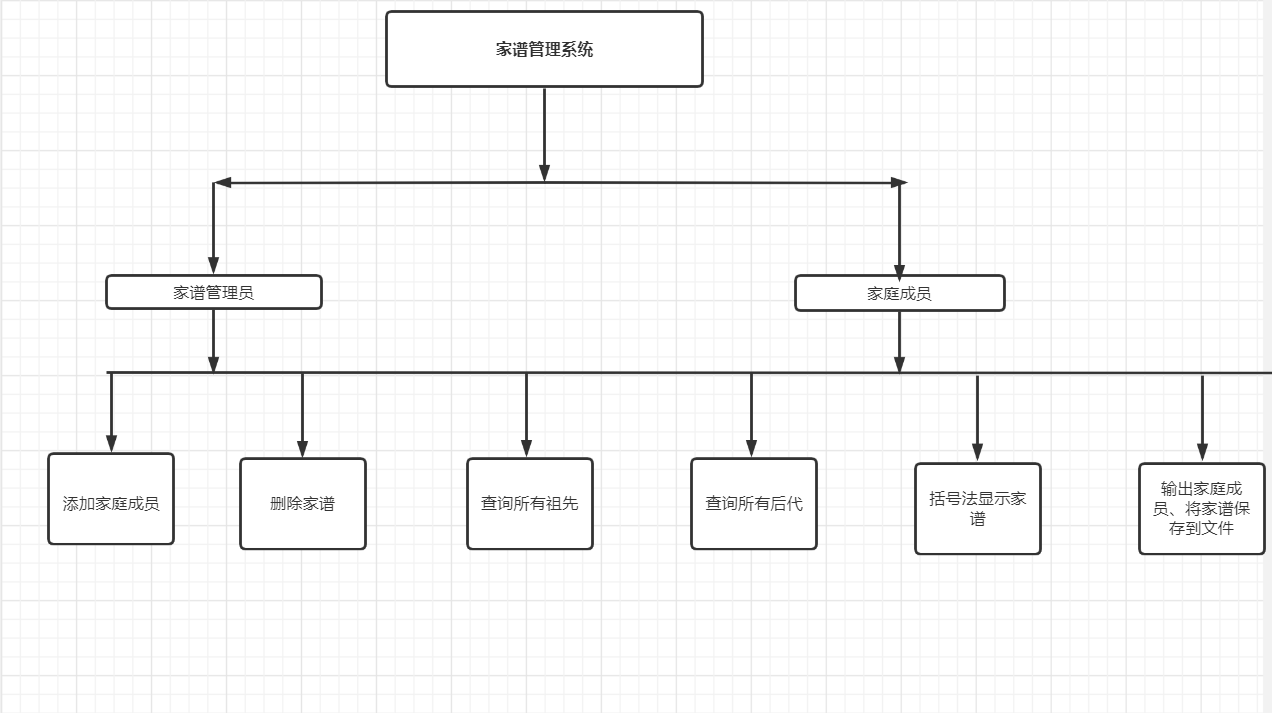


图2-1 系统结构图

**2.4 系统子程序及功能设计**

本系统的各个功能模块共涉及13个函数。

**2.4.1相关函数**

|  |  |
| --- | --- |
| 函数名称 | 功能简述 |
| TreeNode \*savePerson() | //输入存入父母孩子 |
| TreeNode\*Find(TreeNode \*node,string name) | //遍历查找人名 |
| Void save(TreeNode \*root) | //存小孩父母 如果同一父母 |
| VoidshowFamily(TreeNode \*node) | //家谱记录的输出 |
| void freeTree(TreeNode \*node) | //清除全部文件记录 释放树 |
| void destroyFile() | //清除文件完成 |
| void saveInFile(TreeNode \*node) | //将家谱记录存盘  // 树保存到文件夹 |
| void readFile() | //从文件读树 |
| TreeNode\* BuildTree() | //建树 |
| void printBinaryTree(TreeNode \*node) | //用括号表示法输出家谱二叉树 |
| void ShowSon(TreeNode \*node,string name) | //显示查询的所有孩子 |
| bool ShowAllFather(TreeNode \*node,string name) | //查找某人的所有祖先（这里的祖先是指所设计的二叉树结构中某结点的所有祖先结点） |
| int main() | //主函数 |

表2-1 乘客相关函数

**3 运行环境**

硬件环境： PC机内存：8G 硬盘：512G

软件环境： 操作系统：windows11

**4 开发工具和编程语言**

开发环境：Dev-C++

编程语言：C++语言

**5 详细设计**

在概要设计的基础上，对每个模块进行内部逻辑处理部分详细设计。分析主要功能实现的要求和主要代码的详细介绍。程序主要分为两大功能。

（1）文件操作功能：家谱记录的输入，家谱记录的输出，清除全部文件记录和将家谱记录存盘。要求在输入家谱记录时按从祖先到子孙的顺序输入，第一个家谱记录的父亲域为所有人的祖先。

（2）家谱操作功能：用括号表示法输出家谱二叉树，查找某人的所有儿子，查找某人的所有祖先（这里的祖先是指所设计的二叉树结构中某结点的所有祖先结点）。树的基本结构如图所示:

\* 父

\* /

\* 母

\* \

\* 子1

\* / \

\* 媳1 子2

\* / \

\* 媳2 子3

\* /

\* 媳3

**5.1 添加家谱模块**

在程序的最开始，会询问用户创建新的家谱或者是从事先保存的文件中读取家谱信息，这是会分为两种情况。第一种情况对于大部分新用户没有事先保存的家谱，都会新建一个家谱进行家庭成员的输入。

if(a=='1'){

ancestor = savePerson(); ;

break;

}

else if(a=='2'){

fstream f("family.txt");

f.seekg(0,ios::end);

streampos fp = f.tellg();

if(int(fp)==0){

cout<<"文件为空"<<endl;

}else{

readFile();

ancestor = BuildTree();

break;

}f.close();

家谱记录的输入(要求在输入家谱记录时按从祖先到子孙的顺序输入，第一个家谱记录的父亲域为所有人的祖先。)依次输入父母孩子名字，以空格分开。

TreeNode \*savePerson(){

cout<<"输入父母孩子名字，以空格分开:";

string s1,s2,s3;

cin>>s1>>s2>>s3;

TreeNode \*father = new TreeNode;

father->lChild = father->rChild = NULL;

father->name = s1;

对于母亲和孩子的输入同理，最后返回return father;

第二种情况是之前使用过程序，并保存的有家谱信息。可以选择用文件中存储家谱进行家庭成员的输入添加。

void readFile(){ //把文件中的树节点读进string数组

ifstream f("family.txt",ios::in);

int i = 0;

while(!f.eof()){

f>>s[i];

i++;

}

f.close();

}

是家谱中已有祖先的情况下，父是根节点，没老婆儿子 result是在家谱找到的父节点；有妻子儿子，指向父老婆（女）；若存入的孩子父母并不是同一父母，则输出"没有找到你的祖先,不存入"。

void save(TreeNode \*root){

TreeNode \*fam = savePerson();//父是根节点

TreeNode \*result = Find(root,fam->name);

在result！= NULL的前提下：

if(result->lChild==NULL){

result->lChild = fam->lChild;

}else{

TreeNode \*temp = result->lChild;

while(temp->rChild!=NULL){

temp = temp->rChild;

}

temp->rChild = fam->lChild->rChild; //fam中的孩子

在result == NULL的前提下：

cout<<"没有找到你的祖先,不存入"<<endl;

**5.2 删除家谱模块**

此模块将家谱信息会全部删除，故请使用者慎重选择使用，以免误删造成家庭成员数据的丢失，产生不必要的麻烦。该模块通过void freeTree(TreeNode \*node)函数释放树的方式将全部文件记录清除。先释放节点lChild，后释放节点rChild，再将节点删除，赋值为空NULL。

if(node==NULL)return;

freeTree(node->lChild);

freeTree(node->rChild);

delete(node);

node=NULL;

之后将保存的家谱信息的文件一并删除。这个操作需要用到文件读写操作fstream，对文件的操作是通过**stream**的子类**fstream** (**file stream**)来实现的，所以，要用这种方式操作文件，就必须加入头文件**fstream**.h。但是我在原文件的开头写下了#include<bits/stdc++.h>，包含了目前c++所包含的所有头文件。使用起来非常方便。

void destroyFile(){

fstream file("family.txt",ios::out);

cout<<"清除文件完成"<<endl;

}

**5.3 查询家谱模块**

用户可通过此模块选择查找某人的所有祖先，也可以查找某人的所有儿子。通过遍历查找人名。分为两块：

1. 查找某人所有的儿子，这里要判断父亲还是母亲 父亲肯定有左孩子，母亲肯定没左孩子。

TreeNode \*parent = Find(node,name);

if(parent==NULL){cout<<"查无此人"<<endl;return;}

TreeNode \*temp = NULL;

if(parent->lChild==NULL){

temp = parent->rChild;//指向第一个孩子

cout<<name<<"的孩子是：";

while(temp!=NULL){

cout<<temp->name<<" ";

temp = temp->rChild;

}

cout<<endl;

}else if(parent->lChild!=NULL){

temp = parent->lChild->rChild;

cout<<name<<"的孩子是：";

while(temp!=NULL){

cout<<temp->name<<" ";

temp = temp->rChild;

}

cout<<endl;

}

1. 查找某人所有的祖先，这里的祖先是指所设计的二叉树结构中某结点的所有祖先结点。

bool ShowAllFather(TreeNode \*node,string name){

if(node==NULL)return false;

if(node->name==name)return true;

if(ShowAllFather(node->lChild,name)||ShowAllFather(node->rChild,name)){

cout<<node->name<<" ";

return true;

}

return false;

}

**5.4 显示家谱模块**

用户通过此模块可以简洁明了的看出家谱中所有家庭成员的信息，同样有两种方法，一是直接显示所有家庭成员的名字，二是选择用括号表示法输出所有家庭成员的名字，保存的文件是以括号表示法进行输出。

if(node==NULL)return;

if(node->lChild!=NULL){

cout<<node->name<<" ";

TreeNode \*temp = node->lChild;

while(temp!=NULL){

cout<<temp->name<<" ";

temp = temp->rChild;

}

cout<<endl;

TreeNode \*son = node->lChild->rChild;

while(son!=NULL){

showFamily(son);

son = son->rChild;

}

用括号表示法输出家谱二叉树，若根节点存在就输出，不存在则返回；若左右子树存在任一就输出，左右子树都不存在则返回。

if(node->lChild!=NULL||node->rChild!=NULL)cout<<'(';

else return;

if(node->lChild){

printBinaryTree(node->lChild); //左子树存在

} if(node->rChild){

cout<<',';

printBinaryTree(node->rChild);

} cout<<')';

}

**5.5 保存家谱模块**

此模块将用户输入的家庭成员数据进行保存，将家谱记录存盘，树保存到文件夹，其中ofstream默认是以输出方式打开文件，若文件不存在，则自动创建对程序来说 把文件导入程序 就是以输入方式打开文件，把内容写到文件中就是以输出方式打开文件。

void saveInFile(TreeNode \*node){

ofstream of("family.txt",ios::app);

if(node==NULL){

of<<"NULL"<<" ";

return;

}

of<<node->name<<" ";

of.close();

saveInFile(node->lChild);

saveInFile(node->rChild);

}

**6 运行实现**

运行系统，选择文件中存储的家谱，如图6-1所示：

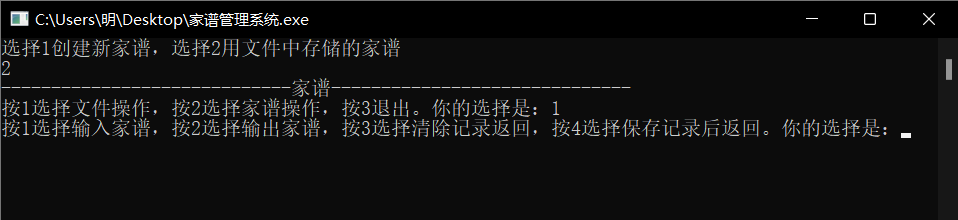


图6-1

输入输出家谱，如图6-2所示：

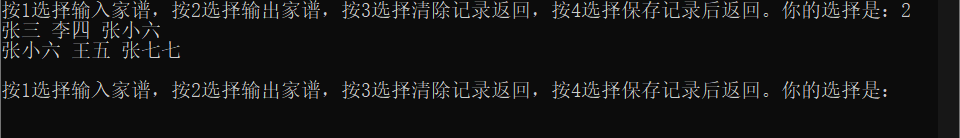
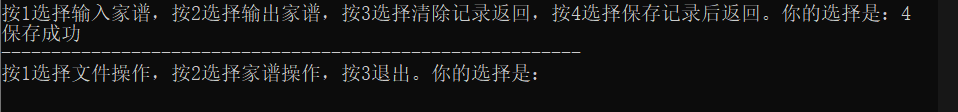


图6-2

保存家谱信息到文件，如图6-3所示：



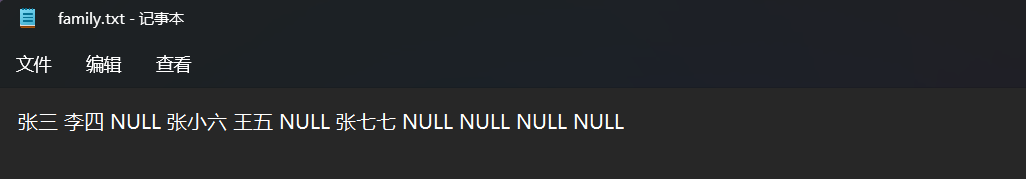


图6-3

括号法输出家谱，如图6-4所示：

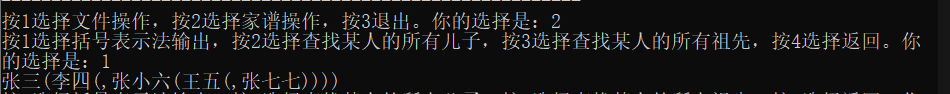


图6-4

查找某人的儿子，如图6-5所示：



图6-5

查找某人的祖先，如图6-6所示：

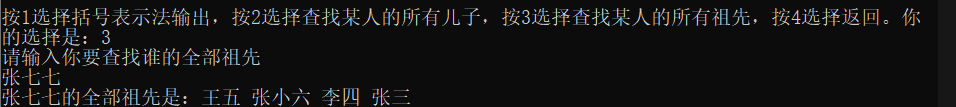


图6-6

输入与祖先姓名不一，如图6-7所示：

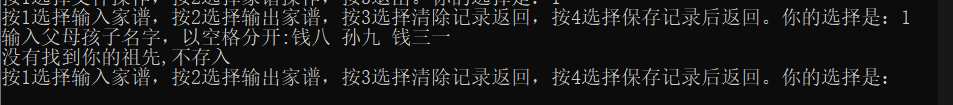


图6-7

此时可以删除现有的家谱，创建一个新的家谱。

删除现有家谱记录，如图6-8所示：

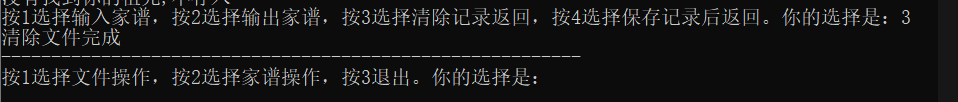


图6-8

创建新家谱并输入输出，如图6-9所示：

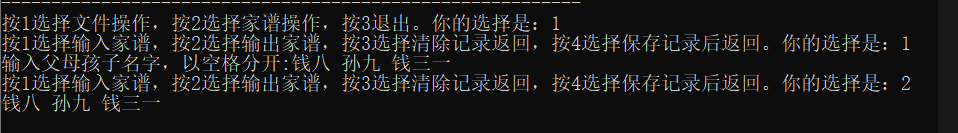


图6-9

该系统运行所有界面，如图6-10所示：

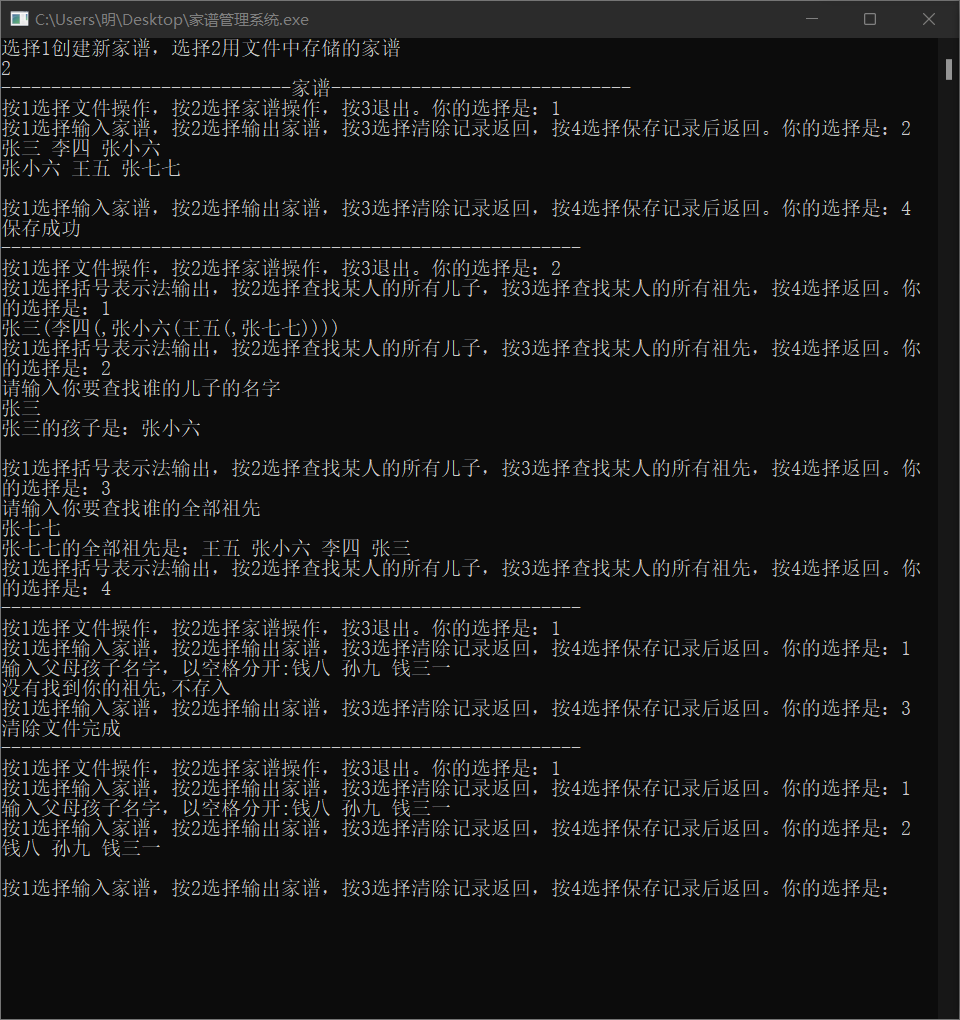


图6-10

**7 心得体会**

本次以家谱管理系统为题的课程设计所用到的知识完全是上学期的知识，通过这次课程设计，我认识到了我对数据结构这门课的掌握程度。

首先我这个课程设计是关于二叉树的，由于是刚接触二叉树，所以我掌握的长度并不深。在编程之前我把有关于二叉树的知识有温习了一遍，还好并没有忘掉。二叉树这章节难度中上等，而且内容广泛，所以我只掌握了百分之六七十。我更加扎实地掌握了有关数据结构方面的知识。在构建完系统的整体框架后，我对每个模块进行了完善和修改，在一些边缘特殊情况的数据的测试时出遇到了许多困难，但经过不断的尝试和搜索资料，我都找到了问题所在并解决了问题。该系统在这个过程中，我学习到了很多新的知识，增强了实践能力，丰富了经验。

然后，在编程中我认识到了自己动手能力的不足，虽然相比较大一而言进步很大，但是我还是不满意，有的在编程中必须看书才能写出来，有的靠百度，很少是自己写的。还好，我自己组装程序的能力还行，要不这东拼西凑的程序根本组装不了。在编程中我还认识到了，编程不能停下，如果编程的时间少了，知识忘的会很快，而且动手也会很慢。同时，同学之间的合作也很重要，每个人掌握的知识都不一样，而且掌握程度也不一样，你不会的别的同学会，所以在大家的共同努力下，编程会变得很容易。在这次编程中，我了解到了自己某些方面的不足，比如说链表的知识，虽然我能做一些有关于链表的编程，但是很慢，没有别人编程的快，另外，二叉树和图的知识最不好掌握，这方面的知识广泛而复杂。

总之，在完成课程设计的过程中，我认识到了自己对知识点的掌握程度还有待提高，也领会了要把所学的理论知识与实际结合起来去解决问题，从理论中得出结论，才能提高自己的实际动手能力、独立思考能力。课设完成的也许不是很完美，但我相信经过不断努力我的编程能力一定越来越强。

**8 参考文献**

[1] 严蔚敏.《数据结构（C语言版）》[M]. 北京：清华大学出版社, 2018.

[2] 谭浩强.《C语言程序设计》[M]. 北京：清华大学出版社, 2018

[3]徐慧民等.C++大学基础教程[M].北京：人民邮电出版社，2005.

[4]王红梅，胡明，王涛.数据结构（C++版）.北京：清华大学出版社，2011.

[5] 苏仕华.数据结构课程设计. 机械工业出版社