

# 杭州电子科技大学

HANGZHOU DIANZI UNIVERSITY

## 数据结构课程设计报告

(2021-2022-2 学期)



题目	家庭族谱的构建			
学院	理学院			
专业	信息与计算科学			
组号	第十六组	学号	姓名	分工
		20071226	童繁	流程图
		20071227	王瀚功	测试
		20071228	王赛豪	文案
		20071229	吴政豪	调试
		20071230	武琦	代码
时间	2022 年 4 月 28 日			

# 目录

1. 需求分析.....	1
2. 项目亮点.....	1
3. 概要设计.....	1
4. 详细设计.....	3
4.1 定义.....	3
4.2 重要函数.....	6
4.2.1 添加成员.....	6
4.2.2 删除成员.....	8
4.2.3 修改成员信息.....	10
5. 用户手册.....	12
5.1 界面.....	12
5.2 删除成员.....	12
5.3 添加成员.....	12
5.4 查找成员.....	13
5.5 修改成员信息.....	13
6. 心得体会.....	13
7. 附录 .....	14
7.1 definition.h .....	14
7.2 main.c .....	15
7.3 tree.c .....	16
7.4 function.c.....	19

## 1. 需求分析

- (1) 利用二叉树的存储结构，使用孩子兄弟表示法，将本人的家谱进行存储、查询和显示等功能。
- (2) 输入：根据姓名和其父名字，插入族谱中相应的位置。
- (3) 查询：根据输入的姓名，查找其在该族谱中属于第几代。
- (3) 输出：以可视化的方式对族谱进行输出。

## 2. 项目亮点

- (1) 建立了家谱的初始化和保存机制；
- (2) 利用了二叉树的存储结构，使用孩子兄弟表示法完成了家谱管理系统；
- (3) 以凹入表的形式打印家谱；
- (4) 设计了成员信息合法的检查函数；
- (5) 添加了删除成员的功能；
- (6) 添加了修改成员信息的功能。

## 3. 概要设计

族谱记录的是同姓的亲人，一般是男性及其后代。对于出嫁的女性，若其后代随父姓，则不记录在本族家谱内；若随母姓，则记录在本族家谱内。

家谱中成员的信息包括：姓名、性别、配偶姓名、辈分、出生日期、是否在世、父亲或母亲的姓名(本家姓)。在修改成员信息的功能中有“记录过世”这一选项，如果家谱中某位成员刚刚过世，在记录时会弹出：“逝者安息，生者奋然。”

程序运行时，首先初始化，读取家谱文件生成一个孩子兄弟树，用户选择功能执行。用户选择退出时，将更新的家谱保存到文件后结束程序。

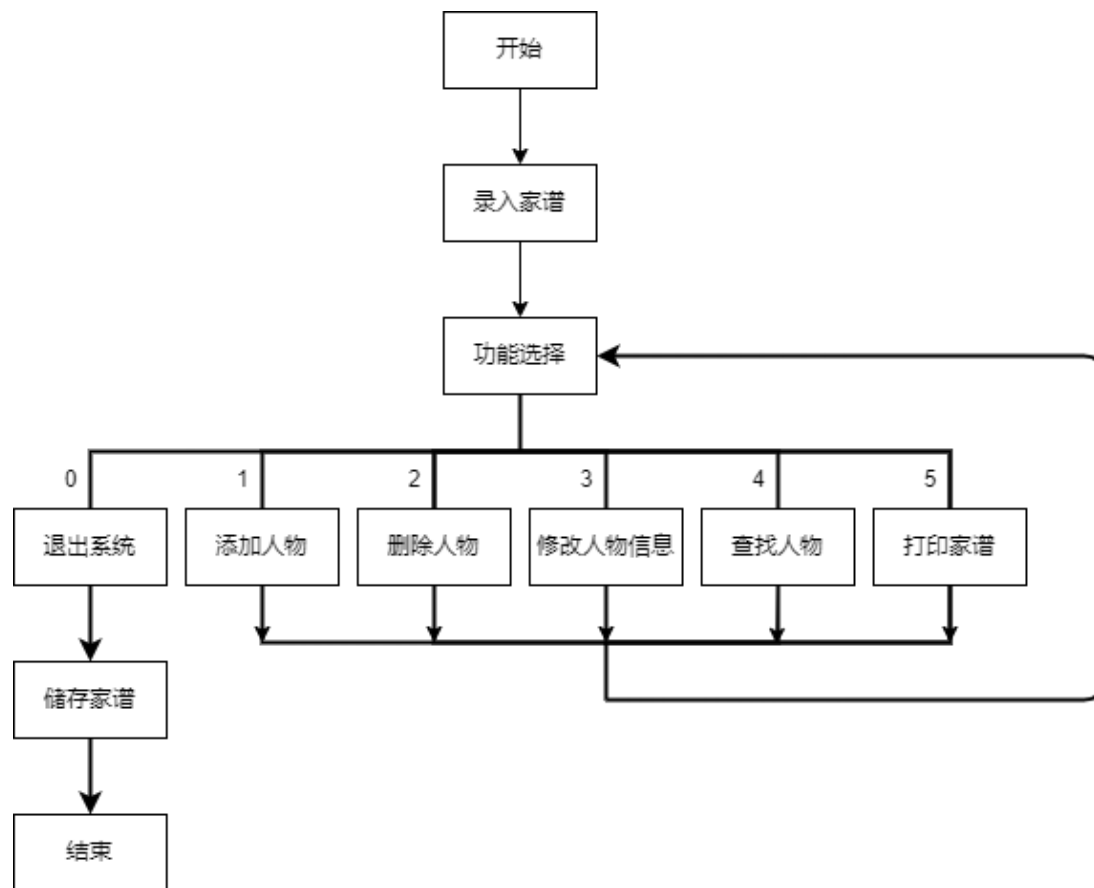


图 1: 主函数流程图

## 4. 详细设计

### 4.1 定义



图 2: 结构函数库

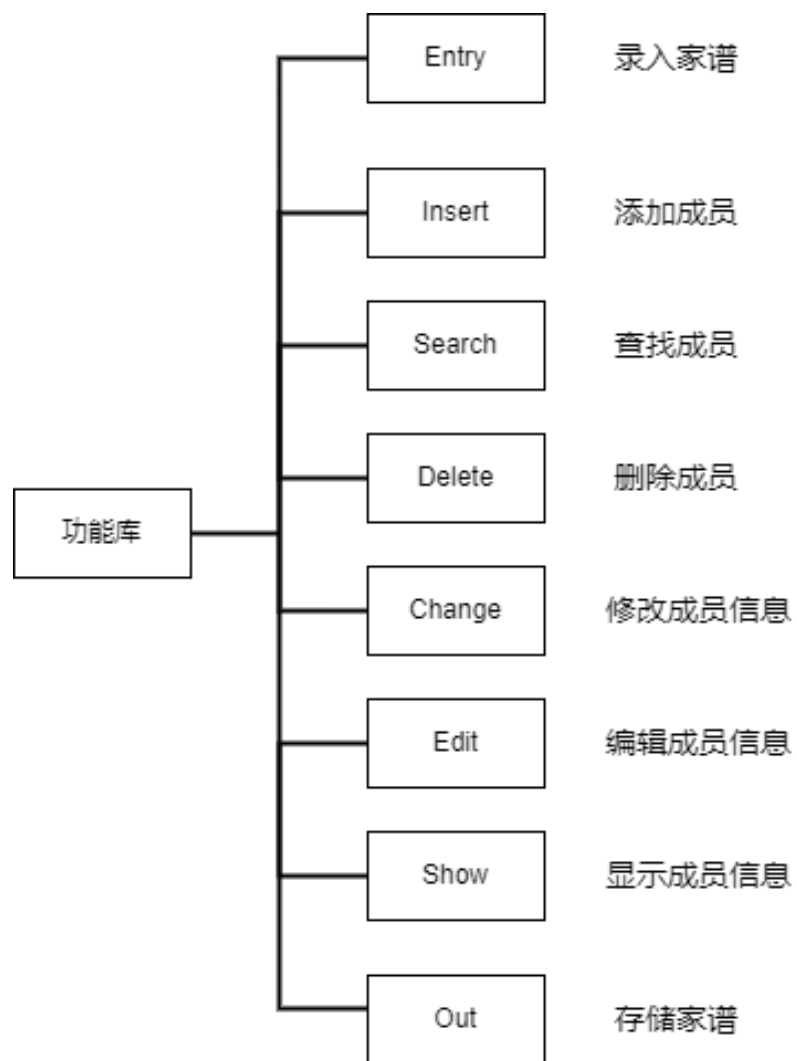


图 3: 功能函数库

```
1 #include<stdio.h>
2 #include<string.h>
3 #include<stdlib.h>
4 #include<malloc.h>
5
6 typedef enum status
7 {
8     TRUE,
9     FALSE,
10    OK,
11    ERROR,
12    SUCCESS,
13    OVERFLOW,
14    EMPTY
15 }Status;//枚举类型返回值
16
17 typedef struct Member
```

```
18 {
19     char name[20]; //姓名
20     int sex; //性别:1为男性 0为女性
21     char spouse[20]; //配偶姓名
22     int seniority; //辈分
23     int birthday; //生日
24     int alive; //是否在世:1为在世 0为过世
25 }Member; //每个家庭成员的信息库
26
27 typedef struct TNode
28 {
29     Member data;
30     struct TNode *firstChild,*nextSibling,*father;
31 }TNode,*Tree,*Forest; //二叉树结构——孩子兄弟表示法
32
33 FILE *fp;
34 Forest T;
35 Tree A,B;
36 //二叉树函数
37 Status InitTree(Tree *A); //创建家谱树
38 Status SearchTree(Forest T,char name[20],Tree *B); //搜索成员姓名
39 Status InsertTree(Tree *father,Tree *child); //插入新成员
40 Status DeleteTree(Tree *A); //删除成员
41 Status DestroyTree(Tree *A); //销毁树
42 Status PrintTree(Tree T); //输出家谱
43 Status SaveTree(Tree T); //保存树
44 Status CheckBirthday(int birthday); //判断生日合法性
45 Status Birthday(int birthday); //转换为生日格式
46 //功能函数
47 Status Entry(); //录入家谱
48 Status Insert(); //添加成员
49 Status Search(); //查找成员
50 Status Delete(); //删除成员
51 Status Change(); //修改成员信息
52 Status Edit(Tree *B,Forest T); //修改成员信息
53 Status Show(Tree *A); //显示成员信息
54 Status Out(); //存储家谱
```

## 4.2 重要函数

### 4.2.1 添加成员

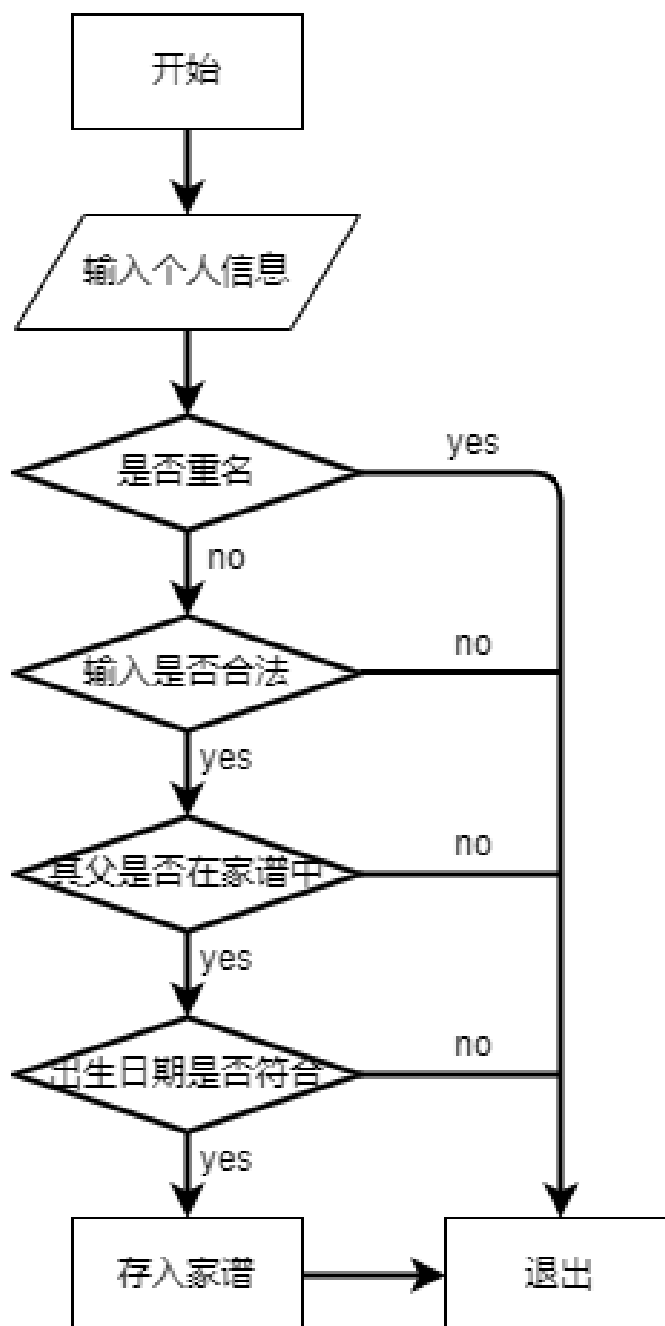


图 4: 添加成员

Listing 1: Insert

```
1 Status Insert()//添加成员
2 {
3     char father[20];
```



```
4   A=NULL;B=NULL;
5   InitTree(&A);
6   printf("请输入:姓名(不能重名) 性别(男1|女0) 配偶姓名(没有输入无) 出生日期(8位数)
      是否在世(1|0) 父亲姓名\n ");
7   scanf("%s %d %s %d %d
          %s",&A->data.name,&A->data.sex,&A->data.spouse,&A->data.birthday,&A->data.alive,father);
8   SearchTree(T,A->data.name,&B);
9   if(B!=NULL)
10  {
11      printf("姓名与家族成员重名!\n");
12      free(A);
13      A=NULL;
14      return ERROR;
15  }
16  B=NULL;
17  if(A->data.sex!=1&&A->data.sex!=0)
18  {
19      printf("性别输入有误!\n");
20      free(A);
21      A=NULL;
22      return ERROR;
23  }
24  if(A->data.alive!=1&&A->data.alive!=0)
25  {
26      printf("莫把生命当儿戏!\n");
27      free(A);
28      A=NULL;
29      return ERROR;
30  }
31  if(!CheckBirthday(A->data.birthday))
32  {
33      printf("出生日期输入有误!\n");
34      free(A);
35      A=NULL;
36      return ERROR;
37  }
38  SearchTree(T,father,&B);
39  if(B==NULL)
40  {
41      printf("此人的父亲不在家谱里!\n");
42      free(A);
43      A=NULL;
44      return ERROR;
45  }
46  if(B->data.seniority!=0&&B->data.birthday>=A->data.birthday)
47  {
48      printf("出生日期不能比父亲早!\n");
```

```
49     free(A);
50     A=NULL;
51     B=NULL;
52     return ERROR;
53 }
54 InsertTree(&B,&A);
55 printf("添加了新的家庭成员!\n");
56 A=NULL;
57 return OK;
58 }
```

4.2.2 删除成员

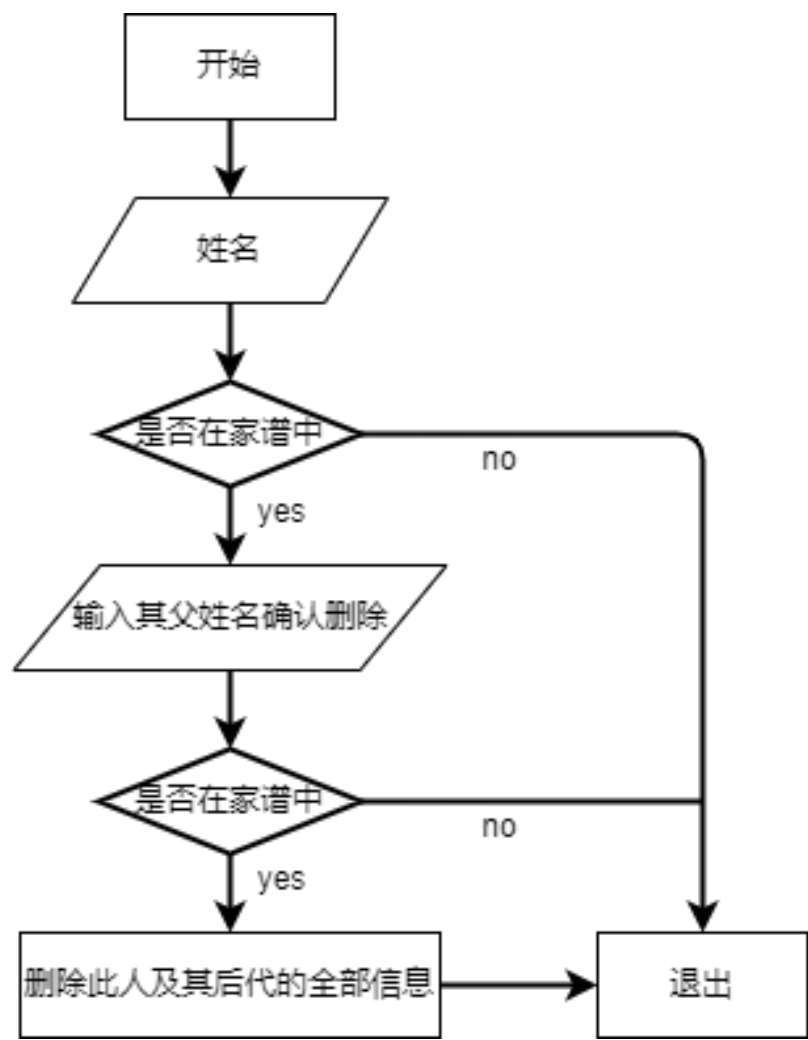


图 5: 删除成员

Listing 2: Delete

```
1 Status Delete()//删除成员
```

```
2  {
3      A=NULL;B=NULL;
4      char name[20],fatherName[20];
5      printf("请输入姓名:");
6      scanf("%s",name);
7      SearchTree(T,name,&A);
8      if(A==NULL)
9      {
10         printf("不存在名为%s的人\n",name);
11         return ERROR;
12     }
13     printf("请输入其父亲的名字确定删除:");
14     scanf("%s",fatherName);
15     SearchTree(T,fatherName,&B);
16     if(B==NULL)
17     {
18         printf("此人的父亲不在家谱里!\n");
19         return ERROR;
20     }
21     DeleteTree(&A);
22     printf("已删除%s及其后代的全部信息!\n",name);
23     A=NULL;B=NULL;
24     return OK;
25 }
```

## 4.2.3 修改成员信息

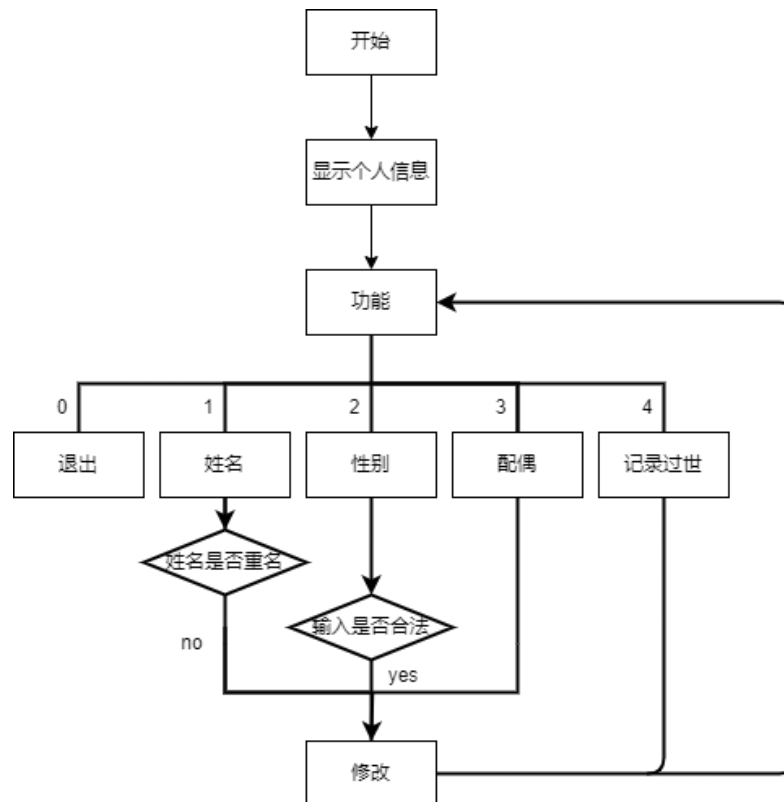


图 6: 修改成员信息

Listing 3: Edit

```

1 Status Edit(Tree *B,Forest T)//修改成员信息
2 {
3     if((*B)==NULL) return ERROR;
4     if((*B)->data.seniority==0)
5     {
6         printf("该节点为森林根结点,禁止操作!\n");
7         system("pause");
8         return ERROR;
9     }
10    int choice;
11    A:system("cls");
12    Show(&(*B));
13    printf("\n您想修改什么信息?\n\n");
14    printf("*****\n");
15    printf("0.退出 1.姓名 2.性别 3.配偶 4.记录过世(慎重选择)\n");
16    printf("*****\n");
17    printf("请输入序号:");
18    fflush(stdin);
19    scanf("%d",&choice);

```

```
20     switch(choice)
21     {
22         case 1:
23             printf("请输入新名字:");
24             char name[20];
25             fflush(stdin);
26             scanf("%s",name);
27             Tree D;
28             SearchTree(T,name,&D);
29             if(D!=NULL) printf("与其他家庭成员重名!\n");
30             else strcpy((*B)->data.name,name);
31             system("pause");goto A;
32         case 2:
33             printf("你的性别为(男1|女0):");
34             int sex;
35             fflush(stdin);
36             scanf("%d",&sex);
37             if(sex!=0&&sex!=1)
38             {
39                 printf("输入有误\n");
40                 system("pause"); goto A;
41             }
42             (*B)->data.sex=sex;
43             system("pause"); goto A;
44         case 3:
45             printf("请输入配偶姓名:");
46             char spouse[100];
47             fflush(stdin);
48             scanf("%s",spouse);
49             strcpy((*B)->data.spouse,spouse);
50             system("pause"); goto A;
51         case 4:
52             if((*B)->data.alive) (*B)->data.alive=0;
53             printf("逝者安息,生者奋然.\n");
54             system("pause"); goto A;
55         case 0:break;
56         default:goto A;
57     }
58     return OK;
59 }
```

## 5. 用户手册

### 5.1 界面

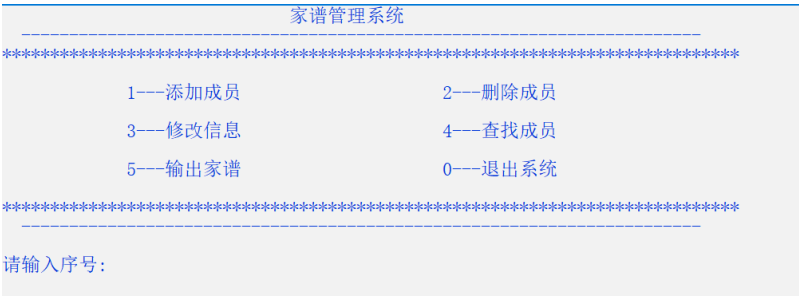


图 7: 界面

### 5.2 删除成员

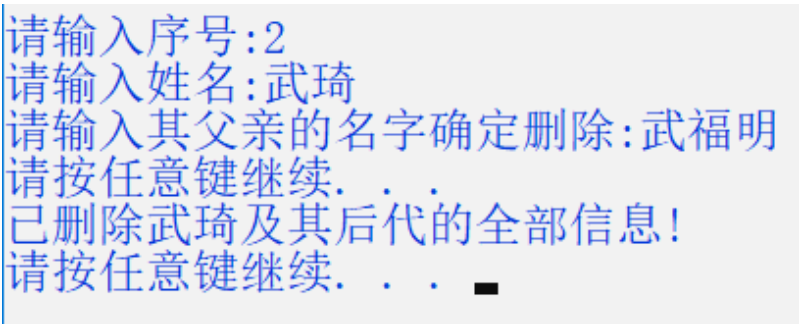


图 8: 删除成员

### 5.3 添加成员

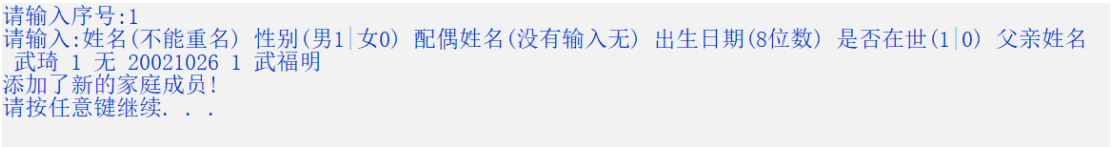


图 9: 添加成员

## 5.4 查找成员

```
请输入序号:4
请输入姓名:武琦
武琦 性别:男 配偶姓名:无 辈分为5 出生日期:2002年10月26日 在世 父亲姓名:武福明
请按任意键继续. . . ■
```

图 10: 查找成员

## 5.5 修改成员信息

```
武琦 性别:男 配偶姓名:无 辈分为5 出生日期:2002年10月26日 在世 父亲姓名:武福明
您想修改什么信息?

*****
0. 退出  1. 姓名  2. 性别  3. 配偶  4. 记录过世(慎重选择)
*****
请输入序号:1
请输入新名字:武璟
与其他家庭成员重名!
请按任意键继续. . . ■
```

图 11: 修改成员信息

## 6. 心得体会

这次课程设计的心得体会通过实践我们的收获如下:

1. 在这次的家庭族谱构建过程中, 我们更深刻地了解了二叉树的特点与用法。
2. 在做一个较大的程序过程中, 应该学会边编写程序边运行, 即完成了一个功能, 也要对其调试, 这样有利于我们高效地完成项目, 并在调试 BUG 的过程可以大大减小难度。
3. 必须要有良好的编程习惯。首先编码风格要统一规范, 这样不仅有利于代码的阅读, 更有利于代码的维护。其次在一些代码方面要细心谨慎, 减少 BUG 出现的机率。
4. 更加系统地学习了 C 语言的 `system` 函数, 对 `goto` 函数的运用更加熟练。

## 7. 附录

### 7.1 definition.h

```
1  #include<stdio.h>
2  #include<string.h>
3  #include<stdlib.h>
4  #include<malloc.h>
5
6  typedef enum status
7  {
8      TRUE,
9      FALSE,
10     OK,
11     ERROR,
12     SUCCESS,
13     OVERFLOW,
14     EMPTY
15 }Status; //枚举类型返回值
16
17 typedef struct Member
18 {
19     char name[20]; //姓名
20     int sex; //性别:1为男性 0为女性
21     char spouse[20]; //配偶姓名
22     int seniority; //辈分
23     int birthday; //生日
24     int alive; //是否在世:1为在世 0为过世
25 }Member; //每个家庭成员的信息库
26
27 typedef struct TNode
28 {
29     Member data;
30     struct TNode *firstChild,*nextSibling,*father;
31 }TNode,*Tree,*Forest; //二叉树结构——孩子兄弟表示法
32
33 FILE *fp;
34 Forest T;
35 Tree A,B;
36 //二叉树函数
37 Status InitTree(Tree *A); //创建家谱树
38 Status SearchTree(Forest T,char name[20],Tree *B); //搜索成员姓名
39 Status InsertTree(Tree *father,Tree *child); //插入新成员
40 Status DeleteTree(Tree *A); //删除成员
41 Status DestroyTree(Tree *A); //销毁树
42 Status PrintTree(Tree T); //输出家谱
```



```

43 Status SaveTree(Tree T);//保存树
44 Status CheckBirthday(int birthday);//判断生日合法性
45 Status Birthday(int birthday);//转换为生日格式
46 //功能函数
47 Status Entry();//录入家谱
48 Status Insert();//添加成员
49 Status Search();//查找成员
50 Status Delete();//删除成员
51 Status Change();//修改成员信息
52 Status Edit(Tree *B,Forest T);//修改成员信息
53 Status Show(Tree *A);//显示成员信息
54 Status Out();//存储家谱

```

## 7.2 main.c

```

1  #include "definition.h"
2  #include "tree.c"
3  #include "function.c"
4
5  Status main()
6  {
7      Entry();
8      int choice;
9      A:system("cls");
10     system("color F1");
11     printf("                家谱管理系统\n");
12     printf(" ----- \n");
13     printf("***** \n\n");
14     printf("          1---添加成员          2---删除成员\n\n");
15     printf("          3---修改信息          4---查找成员 \n\n");
16     printf("          5---输出家谱          0---退出系统\n\n");
17     printf("***** \n");
18     printf("
          ----- \n\n");
19     printf("请输入序号:");
20     fflush(stdin);
21     scanf("%d",&choice);
22     switch(choice)
23     {
24         case 1:Insert();system("pause");goto A;
25         case 2:Delete();system("pause");goto A;
26         case 3:Change();system("pause");goto A;
27         case 4:Search();system("pause");goto A;
28         case 5:PrintTree(T);system("pause");goto A;
29         case 0:break;

```

```
30     default:goto A;
31 }
32 Out(T);
33 system("pause");
34 return OK;
35 }
```

### 7.3 tree.c

```
1  Status InitTree(Tree *A)//创建家谱树
2  {
3      (*A)=(Tree)malloc(sizeof(TNode));
4      if((*A)==NULL) return ERROR;
5      (*A)->father=NULL;
6      (*A)->firstChild=NULL;
7      (*A)->nextSibling=NULL;
8      (*A)->data.birthday=0;
9      (*A)->data.seniority=0;
10     (*A)->data.sex=0;
11     (*A)->data.alive=0;
12     strcpy((*A)->data.name,"无");
13     strcpy((*A)->data.spouse,"无");
14     return OK;
15 }
16
17 Status SearchTree(Forest T,char name[20],Tree *B)//搜索成员姓名
18 {
19     if(T==NULL) return ERROR;
20     if(strcmp(T->data.name,name)==0)
21     {
22         (*B)=T;
23         return OK;
24     }
25     SearchTree(T->firstChild,name,&(*B));
26     if((*B)!=NULL) return OK;
27     SearchTree(T->nextSibling,name,&(*B));
28     return OK;
29 }
30
31 Status InsertTree(Tree *father,Tree *child)//插入新成员
32 {
33     (*child)->data.seniority=(*father)->data.seniority+1;
34     (*child)->father=(*father);
35     int date=(*child)->data.birthday;
36     Tree kid1=(*father)->firstChild,kid2=kid1;
```

```

37     if(kid1==NULL)
38     {
39         (*father)->firstChild=(*child);
40         return OK;
41     }
42     if(kid1->data.birthday>=date)
43     {
44         (*child)->nextSibling=kid1;
45         (*father)->firstChild=(*child);
46         return OK;
47     }
48     for(kid1=kid2->nextSibling;kid1!=NULL&&date>kid1->data.birthday;)
49     {
50         kid2=kid2->nextSibling;
51         kid1=kid2->nextSibling;
52     }
53     (*child)->nextSibling=kid1;
54     kid2->nextSibling=(*child);
55     return OK;
56 }
57
58 Status DeleteTree(Tree *A)//删除成员
59 {
60     Tree kid;
61     if((*A)->father->firstChild==(*A))
62         (*A)->father->firstChild=(*A)->nextSibling;
63     else
64     {
65         for(kid=(*A)->father->firstChild;kid->nextSibling!=(*A);)
66             kid=kid->nextSibling;
67         kid->nextSibling=(*A)->nextSibling;
68     }
69     (*A)->nextSibling=NULL;
70     (*A)->father=NULL;
71     system("pause");
72     DestroyTree(&(*A));
73     return OK;
74 }
75
76 Status DestroyTree(Tree *A)//销毁树
77 {
78     if((*A)==NULL) return ERROR;
79     DestroyTree(&((*A)->firstChild));
80     DestroyTree(&((*A)->nextSibling));
81     free(*A);
82     return OK;
83 }

```

```

84
85 Status PrintTree(Tree T)//输出家谱
86 {
87     if(T)
88     {
89         for(int t=1;t<T->data.seniority;t++) printf(" ");
90         if(T->data.seniority)
91         {
92             printf("[%d]",T->data.seniority);
93             Birthday(T->data.birthday);
94             printf("%s\n\n",T->data.name);
95         }
96         PrintTree(T->firstChild);
97         PrintTree(T->nextSibling);
98     }
99     return OK;
100 }
101
102 Status SaveTree(Tree T)//保存树
103 {
104     if(T)
105     {
106         if(T->data.seniority)
107             fprintf(fp,"%s,%d,%s,%d,%d,%s\n",T->data.name,T->data.sex,T->data.spouse,T->data.birthday,T->data
108             SaveTree(T->firstChild);
109             SaveTree(T->nextSibling);
110     }
111     return OK;
112 }
113 Status CheckBirthday(int birthday)//判断生日合法性
114 {
115     int year,month,day;
116     char months[12]={31,28,31,30,31,30,31,31,30,31,30,31};
117     year=birthday/10000;
118     month=(birthday-year*10000)/100;
119     day=birthday%100;
120     if(month==2) (((year%4==0)&&(year%100!=0))|| (year%400==0))?months[1]+=1:months[1];
121     if(month>12||month<1||day>months[month-1]||day<1) return ERROR;
122     return OK;
123 }
124
125 Status Birthday(int birthday)//转换为生日格式
126 {
127     int year,month,day;
128     year=birthday/10000;
129     month=(birthday-year*10000)/100;

```

```

130     day=birthday%100;
131     printf("%d年%d月%d日",year,month,day);
132     return OK;
133 }

```

## 7.4 function.c

```

1  Status Entry()//录入家谱
2  {
3      InitTree(&T);
4      char *line,*record,buffer[1024];
5      if((fp=fopen("family.csv","r"))==NULL) return ERROR;
6      char father[20];
7      fseek(fp,39L,SEEK_SET);
8      while((line=fgets(buffer,sizeof(buffer),fp))!=NULL)
9      {
10         A=NULL;B=NULL;
11         InitTree(&A);
12         record=strtok(line,",");
13         if(record!=NULL)
14         {
15             sprintf(A->data.name,"%s",record);record=strtok(NULL,",");
16             A->data.sex=atoi(record);record=strtok(NULL,",");
17             sprintf(A->data.spouse,"%s",record);record=strtok(NULL,",");
18             A->data.birthday=atoi(record);record=strtok(NULL,",");
19             A->data.alive=atoi(record);record=strtok(NULL,",");
20             sprintf(father,"%s",record);record=strtok(NULL,",");
21             SearchTree(T,father,&B);
22             if(B!=NULL) InsertTree(&B,&A);
23         }
24     }
25     free(A);
26     A=NULL;
27     fclose(fp);
28     return OK;
29 }
30
31 Status Insert()//添加成员
32 {
33     char father[20];
34     A=NULL;B=NULL;
35     InitTree(&A);
36     printf("请输入:姓名(不能重名) 性别(男1|女0) 配偶姓名(没有输入无) 出生日期(8位数)
           是否在世(1|0) 父亲姓名\n ");
37     scanf("%s %d %s %d %d

```

```
        "%s",&A->data.name,&A->data.sex,&A->data.spouse,&A->data.birthday,&A->data.alive,father);
38  SearchTree(T,A->data.name,&B);
39  if(B!=NULL)
40  {
41      printf("姓名与家族成员重名!\n");
42      free(A);
43      A=NULL;
44      return ERROR;
45  }
46  B=NULL;
47  if(A->data.sex!=1&&A->data.sex!=0)
48  {
49      printf("性别输入有误!\n");
50      free(A);
51      A=NULL;
52      return ERROR;
53  }
54  if(A->data.alive!=1&&A->data.alive!=0)
55  {
56      printf("莫把生命当儿戏!\n");
57      free(A);
58      A=NULL;
59      return ERROR;
60  }
61  if(!CheckBirthday(A->data.birthday))
62  {
63      printf("出生日期输入有误!\n");
64      free(A);
65      A=NULL;
66      return ERROR;
67  }
68  SearchTree(T,father,&B);
69  if(B==NULL)
70  {
71      printf("此人的父亲不在家谱里!\n");
72      free(A);
73      A=NULL;
74      return ERROR;
75  }
76  if(B->data.seniority!=0&&B->data.birthday>A->data.birthday)
77  {
78      printf("出生日期不能比父亲早!\n");
79      free(A);
80      A=NULL;
81      B=NULL;
82      return ERROR;
83  }
```

```
84     InsertTree(&B,&A);
85     printf("添加了新的家庭成员!\n");
86     A=NULL;
87     return OK;
88 }
89
90 Status Search()//查找成员
91 {
92     char name[20];
93     printf("请输入姓名:");
94     scanf("%s",name);
95     A=NULL;
96     SearchTree(T,name,&A);
97     if(A==NULL)
98     {
99         printf("不存在名为%s的人\n",name);
100         return ERROR;
101     }
102     Show(&A);
103     A=NULL;
104     return OK;
105 }
106
107 Status Delete()//删除成员
108 {
109     A=NULL;B=NULL;
110     char name[20],fatherName[20];
111     printf("请输入姓名:");
112     scanf("%s",name);
113     SearchTree(T,name,&A);
114     if(A==NULL)
115     {
116         printf("不存在名为%s的人\n",name);
117         return ERROR;
118     }
119     printf("请输入其父亲的名字确定删除:");
120     scanf("%s",fatherName);
121     SearchTree(T,fatherName,&B);
122     if(B==NULL)
123     {
124         printf("此人的父亲不在家谱里!\n");
125         return ERROR;
126     }
127     DeleteTree(&A);
128     printf("已删除%s及其后代的全部信息!\n",name);
129     A=NULL;B=NULL;
130     return OK;
```

```

131 }
132
133 Status Change()//修改人物信息
134 {
135     A=NULL;
136     char name[20];
137     printf("请输入姓名:");
138     scanf("%s",name);
139     SearchTree(T,name,&A);
140     if(A==NULL)
141     {
142         printf("不存在名为%s的人\n",name);
143         return ERROR;
144     }
145     Edit(&A,T);
146     A=NULL;
147     return OK;
148 }
149
150 Status Edit(Tree *B,Forest T)//修改成员信息
151 {
152     if((*B)==NULL) return ERROR;
153     if((*B)->data.seniority==0)
154     {
155         printf("该节点为森林根结点,禁止操作!\n");
156         system("pause");
157         return ERROR;
158     }
159     int choice;
160     A:system("cls");
161     Show(&(*B));
162     printf("\n您想修改什么信息?\n\n");
163     printf("*****\n");
164     printf("0.退出 1.姓名 2.性别 3.配偶 4.记录过世(慎重选择)\n");
165     printf("*****\n");
166     printf("请输入序号:");
167     fflush(stdin);
168     scanf("%d",&choice);
169     switch(choice)
170     {
171         case 1:
172             printf("请输入新名字:");
173             char name[20];
174             fflush(stdin);
175             scanf("%s",name);
176             Tree D;
177             SearchTree(T,name,&D);

```



```

178         if(D!=NULL) printf("与其他家庭成员重名!\n");
179         else strcpy((*B)->data.name,name);
180         system("pause");goto A;
181     case 2:
182         printf("你的性别为(男1|女0):");
183         int sex;
184         fflush(stdin);
185         scanf("%d",&sex);
186         if(sex!=0&&sex!=1)
187         {
188             printf("输入有误\n");
189             system("pause"); goto A;
190         }
191         (*B)->data.sex=sex;
192         system("pause"); goto A;
193     case 3:
194         printf("请输入配偶姓名:");
195         char spouse[100];
196         fflush(stdin);
197         scanf("%s",spouse);
198         strcpy((*B)->data.spouse,spouse);
199         system("pause"); goto A;
200     case 4:
201         if((*B)->data.alive) (*B)->data.alive=0;
202         printf("逝者安息,生者奋然.\n");
203         system("pause"); goto A;
204     case 0:break;
205     default:goto A;
206 }
207 return OK;
208 }
209
210 Status Show(Tree *A)//显示成员信息
211 {
212     if((*A)==NULL) return ERROR;
213     if((*A)->data.seniority==0)
214     {
215         printf("该节点为森林根结点, 禁止操作!\n");
216         system("pause");return ERROR;
217     }
218     printf("%s ",(*A)->data.name);
219     if((*A)) printf("性别:男 ");
220     else printf("性别:女 ");
221     printf("配偶姓名:%s ",(*A)->data.spouse);
222     printf("辈分为%d 出生日期:",(*A)->data.seniority);
223     Birthday((*A)->data.birthday);
224     if((*A)->data.alive==1) printf(" 在世 ");

```

```
225     else printf(" 逝世 ");
226     if(((A)->data.seniority)!=1) printf("父亲姓名:%s\n",(*A)->father->data.name);
227     else printf("是家族的始祖\n");
228     return OK;
229 }
230
231 Status Out(Tree T)//存储家谱
232 {
233     if((fp=fopen("family.csv","w"))==NULL) return ERROR;
234     fprintf(fp,"姓名,性别,配偶,出生日期,是否在世,父亲\n");
235     SaveTree(T);
236     return OK;
237 }
```