

目录

一、设计任务.....1

二、需求分析.....1

三、系统设计.....??

四、编码实现.....

五、调试分析与运行结果.....

六、课设总结.....

七、谢辞.....

八、参考文献.....

一、设计任务

[illegible]

通讯录管理系统需要完成 XXXXX 的一般性管理工作:

- (1) XXXXX 的录入、增加、修改、删除、查找和顺序输出等功能。
- (2) 每个记录包含姓名、电话号码、住址等个人信息。
- (3) 具备友好的交互界面。

二、需求分析

[illegible]

1.系统用到的数据

日常生活中用到 XXXXXX 的时候，主要是查朋友的电话和地址信息，因此，XXXXXX 中每个联系 XXXXXXXXXXXXXXXXXXXXXXXX 别、电话号码和住址等 XX 需要建立类或结构体来描述，本课设采用的是结构体来描述一个联系人的信息，也称为一条记录。联系人的全部涉及到的属性如下表所示：

表 1 联系人属性表

属性	数据类型	描述
编号	字符串	每个联系人唯一的 ID 信息，不重复
姓名	字符串	XXXXXXXXXXXXXXXXXXXXXXXXXXXX
性别	字符	XXXXXXXXXXXXXXXXXXXXXXXXXXXX
电话	字符串	XXXXXXXXXXXXXXXXXXXXXXXXXXXX
地址	字符串	XXXXXXXXXXXXXXXXXXXXXXXXXXXX

2.系统用到的函数有:

XXXXX 的一般性管理工作主要是包 XXXXXXXXXXXXXXXXXXXX、

删除、输出等功能，因此需要建 XXXXXXXXXXXXXXXXXXXX 的程序模块性、重用和条理性 XXXXXXXXXXXXXXXXXXXX。系统所有 XXXXXXXXXXXX 功能函数如下表所示：

表 2 功能函数表

函数名	功能
创建函数 CreateList	实现 XXXXX 链表的初始化和建立功能；
插入函数	XXXXXXXXXXXXXXXXXXXXXXXXXX;
按名字排序函数	XXXXXXXXXXXXXXXXXXXXXXXXXX;
按编号排序函数	XXXXXXXXXXXXXXXXXXXXXXXXXX;
查找函数	XXXXXXXXXXXXXXXXXXXXXXXXXX;
修改函数	XXXXXXXXXXXXXXXXXXXXXXXXXX;
删除函数	XXXXXXXXXXXXXXXXXXXXXXXXXX;
输出函数	XXXXXXXXXXXXXXXXXXXXXXXXXX;

3.抽象数据类型定义

```
ADT List{
    数据对象：D= { ai|ai∈ElemSet,i=1,2,...,n,n>=0 }
    数据关系：R1 {<ai-1,ai>| ai-1,ai∈D,i=2,...,n}
    基本操作：
        LinkedList CreateList(void);
        初始条件：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        操作结果：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        void InsertNode(LinkedList head,ListNode *p);
        初始条件：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        操作结果：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        void OrderNodeNUM(linklist *&l);
        初始条件：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        操作结果：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        void OrderNodeNAME(linklist *&l);
        初始条件：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        操作结果：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        ListNode *ListFind(LinkedList head);
        初始条件：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        操作结果：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        void DeleteNode(LinkedList head);
        初始条件：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
        操作结果：XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
```

```

void ModifyNode(LinkList head);
初始条件: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
操作结果: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
void printList(LinkList head);
初始条件: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
操作结果: XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
} ADT List

```

三、系统设计

3.1. 系统功能模块图

系统主要有 7 个功能，分别是创建、插入、查找、删除、修改、输出和 XX 进行操作。通过系统操作主菜单界面，可分别调用各函数完成相应功能。

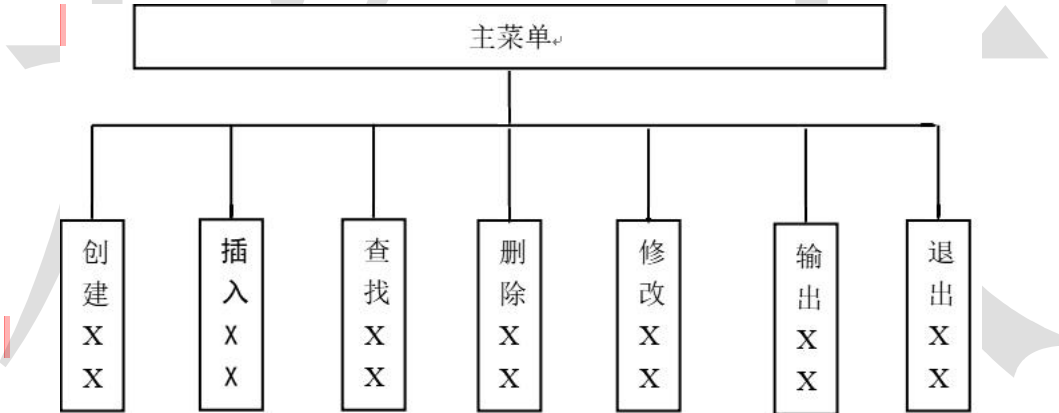


图 1 系统功能模块图

3.2 设计数据元素

由于系统 XX, XX XX XX XX XX, 而 data 数据域又包括每个联系人的基本信息，即编号、姓名、性别、电话号码和住址等信息。

```

struct ElemType{
    char number[5];    //编号

```

]

3.3.1 插入函数

```
void InsertNode(LinkList head,ListNode *p); //实现 XXXXX 结点的插入
```

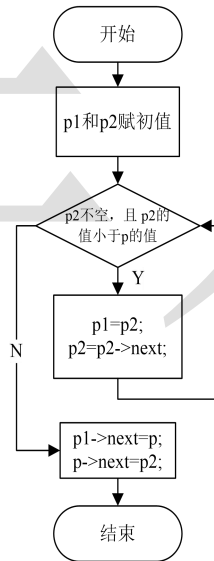
[illegible]

图 2 插入函数流程图

3.3.2 排序函数

```
void OrderNodeNUM(linklist *&l);           //信息按编号排序
```

设计思路与算法描述: 定义三 XXXXXXXXXXXXXXXXXXXXXXXXXXXX 表的头结点, XXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXX

XX
XXXXXXXXXX，采用的是直接插入排序。程序流程图如下所示：

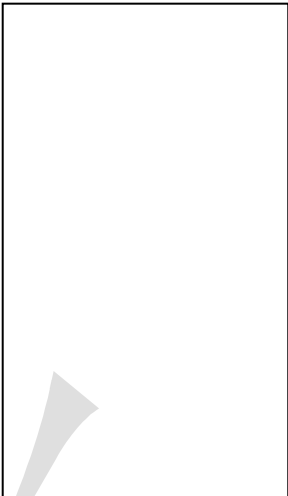


图 3 排序函数流程图

3.3.3 删除函数

```
void DeleteNode(LinkList head);           //实现 XXXXX 结点的删除
```

设计思路与算法描述：定义两个指针 p、q,先利用元素查找函数在链表中
XX
XX
XX
XXXXXXXXXXXXXXXXXXXXXXXXXXXX、free(p)释放被删除的结点空间，删除结束。
程序流程图如下所示：

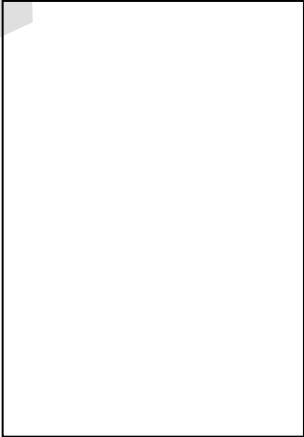


图 4 删除函数流程图

3.3.4 查找函数

```
ListNode *ListFind(LinkList head);           //实现 XXXXX 结点的查找
```

设计思路与算法描述：用户查询元素信息可以有二个选择，一是按每个元素的编号进行查询，另一个XXXXXXXXXXXXXXXXXXXXXXXXXXXX
XXX
XXX
XXX 将上述 p->data.number 改成 p->data.name 即可。程序流程图如下所示：

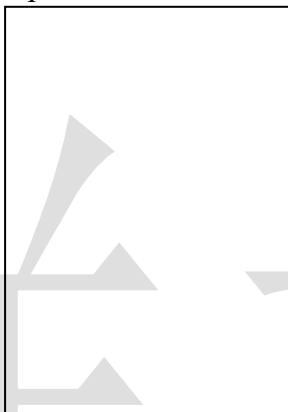
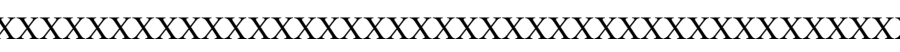


图 5 查找函数流程图

3.3.5 修改函数

```
void ModifyNode(LinkList head);           //实现 XXXXX 结点的修改
```

设计思路与算法描述：这个函数比较简单，只要先利用查询函数确定需XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX信息即可。程序流程图如下所示：



图 6 修改函数流程图

四、编码实现

4.1 XXXXX 建立模块设计

```
LinkedList CreateList(void)
{
    LinkedList head=(ListNode *)malloc(sizeof(ListNode)); //申请头结点
    ListNode *p,*rear;
    int flag=0; //结束指标置0
    rear=head; //尾指针初始化指向头结点
    while(flag==0)
    {
        p=(ListNode *)malloc(sizeof(ListNode)); //申请新结点
        printf("编号(1) 姓名(2) 性别 电话(4) 地址(5)\n");
        printf("-----\n");
        scanf("%s%s%s%s%s", p->data.num, p->data.name, p->data.sex, p->data.phone, p->data.addr);
        rear->next=p; //新结点连接到尾结点之后
        rear=p; //尾指针指向新结点
        printf("结束建表吗? (1/0):");
        scanf("%d",&flag); //读入一个标志数据
    }
    rear->next=NULL; //终端结点指针域置空
    return head; //返回链表头指针
}
```

该模块的时间复杂度是 $O(n)$, 空间复杂度为 $O(n)$ 。

4.2 XXXXX 插入模块设计

```
void InsertNode(LinkedList head, ListNode *p)
{
    ListNode *p1,*p2;
    p1=head;
    p2=p1->next;
    while(p2!=NULL && strcmp(p2->data.num, p->data.num)<0)
    {
        p1=p2; //p1 指向刚访问过的结点
        p2=p2->next; //p2 指向表的下一个结点
    }
}
```

```

    p1->next=p;          //插入 p 所指向的结点
    p->next=p2;          //连接表中剩余部分
}

```

该模块的时间复杂度是 $O(\text{XXXX})$, 空间复杂度为 $O(\text{XXXX})$ 。

4.3 XXXXX 查询模块设计

```

ListNode * ListFind(LinkList head)
{
    //有序 XXXXX 链表上的查找
    ListNode *p; char num[5]; char name[9];
    if(t==1) {
        printf("请输入要查找者的编号: ");
        scanf("%s", num);
        while(p && strcmp(p->data.num, num)<0)
            p=p->next;
        if(p==NULL || strcmp(p->data.num, num)>0)
            p=NULL; //没有查到要查找的通讯者
    } else
        if(t==2) {
            printf("请输入要查找者的姓名: ");
            scanf("%s", name);
            while(p && strcmp(p->data.name, name)!=0)
                p=p->next;
        }
    return p;
}

```

该模块的时间复杂度是 $O(\text{XXXX})$, 空间复杂度为 $O(\text{XXXX})$ 。

4.4 XXXXX 删除模块设计

```

void DelNode(LinkList head)
{
    q=head;
    while(p!=NULL && q->next!=p)
        q=q->next;
    q->next=p->next; //删除结点
    free(p);         //释放被删除的结点空间
    printf("通讯者已被删除! \n");
    return; }

```

}

该模块的时间复杂度是 $O(\text{XXXX})$, 空间复杂度为 $O(\text{XXXX})$ 。

4.5 XXXXXXXXXXXXXXXX 模块设计

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;

```

该模块的时间复杂度是 $O(\text{XXXX})$, 空间复杂度为 $O(\text{XXXX})$ 。

4.6 XXXXXXXXXXXXXXXX 模块设计

```

XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;
XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX;

```

该模块的时间复杂度是 $O(\text{XXXX})$, 空间复杂度为 $O(\text{XXXX})$ 。

五、调试分析与运行结果

5.1 调试分析

在程序调试过程中,遇到了XXXXXXXXXXXXX问题,比较难的有如下几个:

[illegible]

2. XX
 XXXXXXXX 问题；问题的原因是 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX；修改了 XX
 XX 之后，调整 XXXX
 XXXXXXXXXXXXXXXX，最终问题得到了解决。

[illegible][illegible][illegible]

5.2 主菜单

主菜单提供了操作交XXXXXXXXXXXXXXXXXXXXX 反复输入，直至退出系统为止。程序开始界面，如下图所示。

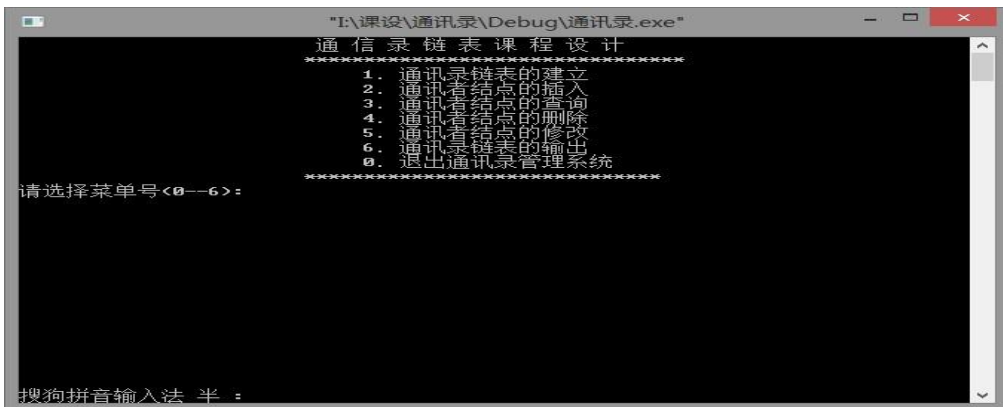


图 7 程序开始界面

5.3 XXXXX 链表的建立

在开始菜单选择 1, 进入链表建立模块; 然 XXXXXXXXXXXXXXXXXXXX 话号 XXXXXXXXXXXXXXXXXXXX 输入结束后, XXXXXXXXXXXXXXXXXXXX 选择 n 退回主界面。操作界面如下图所示。



图 8 建立链表界面

5.4 通讯者信息的插入

在开始界面选 XXXXXXXXXXXXXXXXXXXX。操作界面如下图所示:

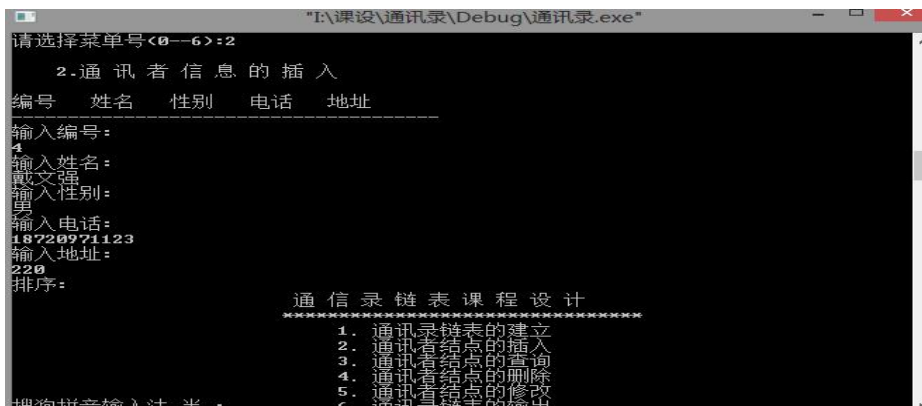


图 9 插入信息界面

5.5XXXXX 信息的查询

在开始菜 XXXXXXXXXXXXXXXXXXXX 表中节点的信息。可以 XXXXXXXX 号查 XXXXXXXXXXXXXXXXXXXX 找。操作界面如下图所示：

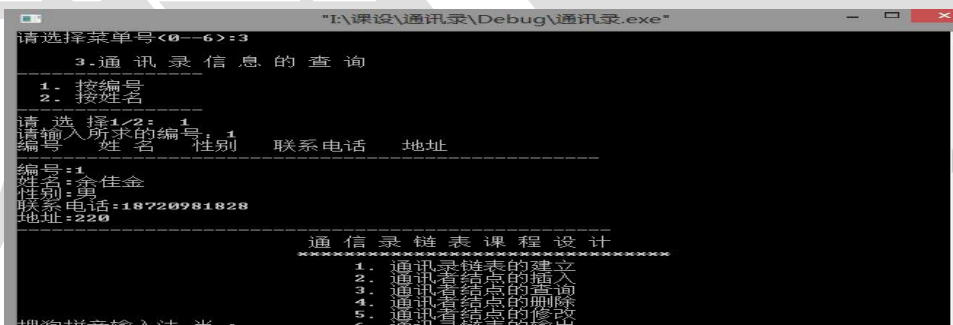


图 10 查询信息界面

5.6XXXXX 信息的删除

在开始菜单选择 4, XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX 的节点。操作界面如下图所示：



图 11 删除界面

5.7XXXXX 信息的修改

在开始菜单选择 5，修 XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX 信息。操作界面如下图所示：



图 12 修改界面

5.8XXXXX 链表按姓名字母顺序输出

在开始菜单选择 6，XXXXXXXXXXXXXXXXXXXXXXXXXXXXX 息。操作界面如下图所示：



图 13 顺序输出界面

六、课设总结

通过这次课程设计，让我学到了很多以前 XXXXXXXXXXXXX 新知识，也让我对 XXXXXXXXXXXXX 新的了解。想要设计一个完整的系统，首先需要需求分析，对题目 XXXXXXXXXXXXXXXXXXXXXXXXXXXX 理解，其次就是功能设

