**数据结构与算法课程设计报告**

题目XXXXX

学号：19201XXX

姓名：XXX

指导老师：罗新 李孝忠 赵青 王林

时间：2021年12月17日

1. 题目描述：

## 学生通讯录管理系统

二、需求分析：

## 通过用户的输入，实现通讯录的增删改查。

1. 题目设计

创建单链表ADT实现内存的管理。通过增加删除修改和查找链表节点来实现通讯录信息的操作。

四、模块设计

// 程序主函数

Int main()

// 链表初始化

int init(LinkList \*head)

// 链表修改

int modify(LinkList head)

// 链表增加

int add(LinkList head)

// 链表删除

int del(LinkList head)

// 链表查询

int query(LinkList head)

// 列出链表

int list(LinkList head)

// 保存到文件

int save(LinkList head)

// 输出环境信息

int welcome\_menu(char \*choice, LinkList head)

// 加载文件函数

int load(LinkList head)

1. 主要源代码清单

主函数代码如下

// 主函数

int main()

{

LinkList head = NULL;

char choice = '\0';

init(&head);

load(head);

welcome\_menu(&choice, head);

return EXIT\_SUCCESS;

}

链表初始化代码如下

int init(LinkList \*head)

{

// 分配内存

\*head = (LinkList)malloc(sizeof(Node));

// 初始化

strcpy((\*head)->number,"\0");

strcpy((\*head)->name,"\0");

strcpy((\*head)->QQ,"\0");

(\*head)->next = NULL;

return 0;

}

链表的修改

int modify(LinkList head)

{

Node \*cur = NULL;

char m\_name[NAME\_LEN];

printf("请输入要修改的联系人姓名:\n");

scanf("%s", m\_name);

// 遍历链表

cur = head->next;

while (cur != NULL) {

if (0 == (strcmp(m\_name, cur->name))) {

printf("请重新输入姓名:");

scanf("%s", cur->name);

printf("请重新输入电话号码:");

scanf("%s", cur->number);

printf("请重新输入QQ号码:");

scanf("%s", cur->QQ);

printf("修改成功!\n");

return 0;

}

cur = cur->next;

}

printf("未找到联系人:%s\n", m\_name);

return 0;

}

链表的添加

int add(LinkList head)

{

Node \*newnode = NULL;

if (!(newnode = (Node \*)malloc(sizeof(Node)))) {

perror("malloc");

}

printf("请输入联系人姓名:\n");

scanf("%s", newnode->name);

printf("请输入联系人号码:\n");

scanf("%s", &newnode->number);

printf("请输入联系人QQ号码:\n");

scanf("%s", &newnode->QQ);

if (strlen(newnode->number) == 11) {

/\*头插法\*/

newnode->next = head->next;

head->next = newnode;

} else {

printf("请输入正确的电话\n");

printf("程序即将退出\n");

exit(-1);

}

return 0;

}

链表的查询

int query(LinkList head)

{

Node \*cur = NULL;

char q\_name[NAME\_LEN];

printf("请输入要查询的联系人姓名或号码:\n");

scanf("%s", q\_name);

// 指向头节点的下一个指针

cur = head->next;

// 遍历链表

while (cur != NULL) {

// 找到对应名字的节点

if ((0 == strcmp(q\_name, cur->name)) || (0 == strcmp(q\_name, cur->number))) {

// 输出信息

printf("查询到的联系人信息如下:\n");

printf("\t姓名\t\t号码\t\tQQ\n");

printf("\t%s\t\t%s\t\t%s\n", cur->name, cur->number, cur->QQ);

return 0;

}

cur = cur->next;

}

printf("数据库无相关信息!\n");

return 0;

}

链表的删除

int del(LinkList head)

{

char d\_name[NAME\_LEN];

Node \*cur = NULL;

Node \*pre = NULL;

list(head);

printf("请输入要删除的联系人姓名:\n");

scanf("%s", d\_name);

// 指向头节点的下一个节点

cur = head->next;

pre = head;

// 遍历所有节点，找到要删除的节点

while (cur != NULL) {

if (0 == (strcmp(d\_name, cur->name))) {

printf("将要删除联系人:%s 号码:%s\n", cur->name, cur->number);

pre->next = cur->next;

free(cur);

}

pre = pre->next;

cur = cur->next;

}

printf("删除成功！\n");

return 0;

}

保存文件

int save(LinkList head)

{

Node \*cur = NULL;

FILE \*w\_file = NULL;

// 打开文件

w\_file = fopen("data.txt", "wt");

if (!w\_file) {

perror("fopen");

exit(-1);

}

// 遍历指针，保存所有链表信息到文件

cur = head->next;

while (cur != NULL) {

fprintf(w\_file, "%s\t%s\t%s\n", cur->name, cur->number, cur->QQ);

printf("正在写入数据: %s\t%s\t%s...\n", cur->name, cur->number, cur->QQ);

cur = cur->next;

}

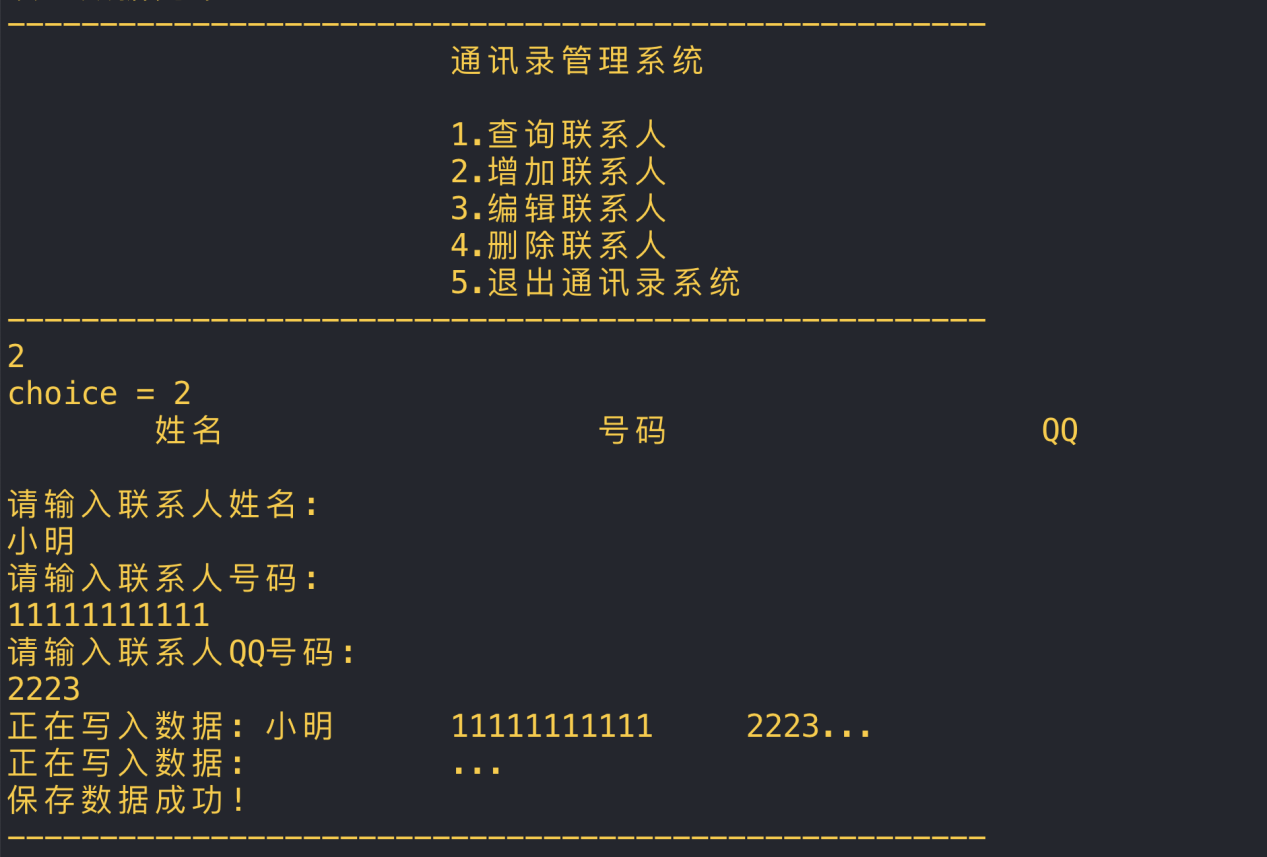
printf("保存数据成功!\n");

return 0;

}

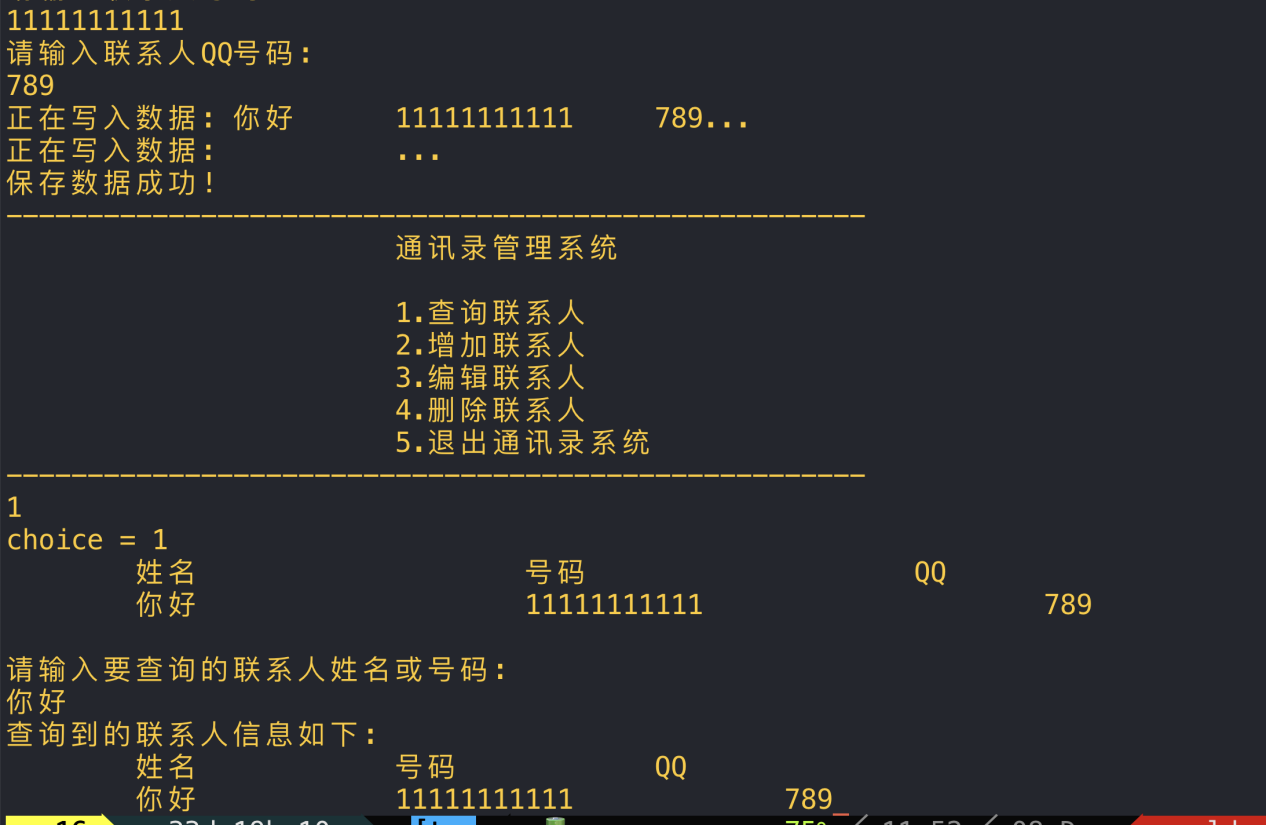
六、测试级运行截图

启动程序

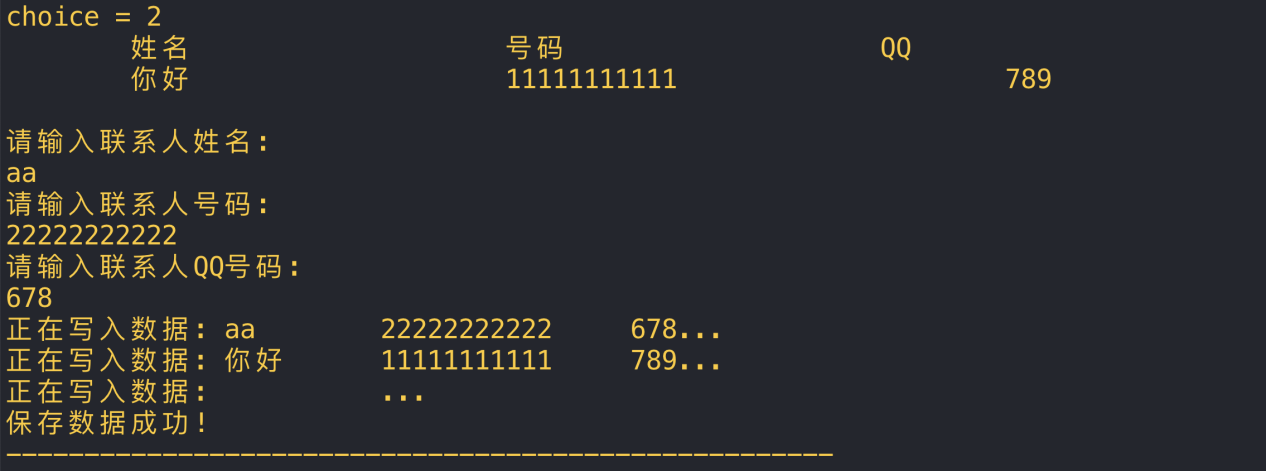


启动的时候会自动打开之前所保存的数据(data.txt)

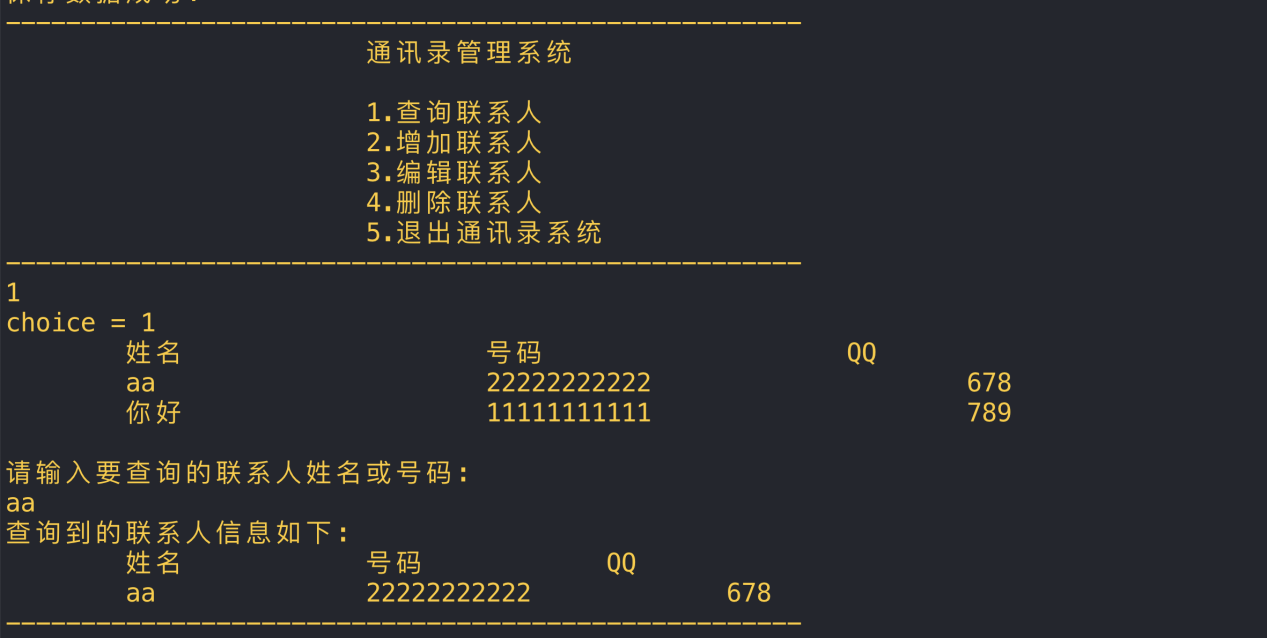
查询信息



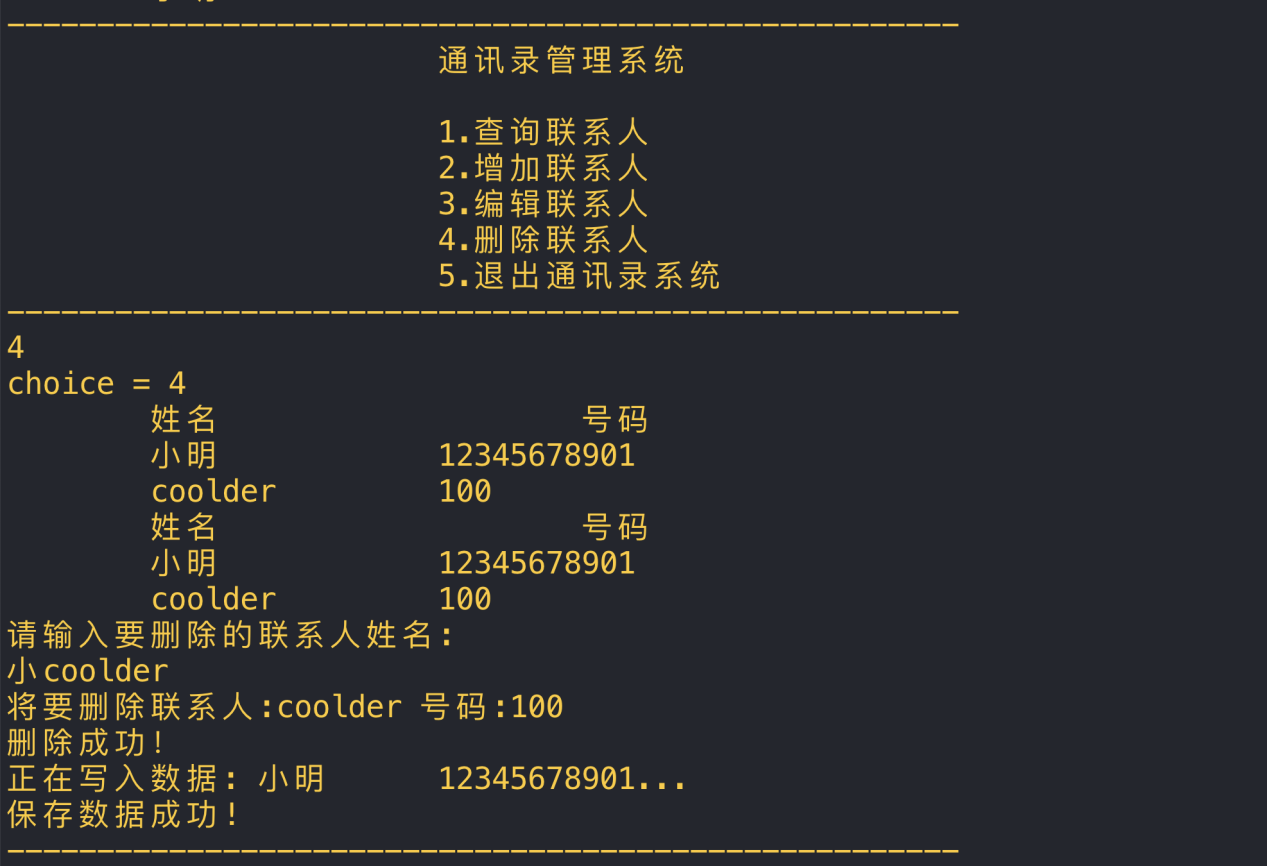
插入通讯录



查询通讯录



删除用户



输入用户名的时候不能输入太长的数据，否则会导致缓冲区溢出最终程序崩溃。

七、用户手册

使用dev c++进行编译并运行，输入数据的时候应当按照需求来输入，如电话号码只能是11位的数字。