通讯录程序制作实验报告

**小组成员**：

* 孔晨皓 P21514131
* 刘天生 P21514109
* 赵永登 P21514093
* 王翔 P21514073
* 董超 P21514087
* 赵泉景 P21514121

**程序介绍**：

有通讯录显示、查找、添加、修改、删除和排序功能的通讯录管理程序

按照以下要求编写程序：

（1）通讯录人数为30。

（2）通讯录信息包括姓名、电话、邮箱、QQ号。

（3）可显示所有信息。

（4）可增加记录，并完成信息的录入。

（5）可根据姓名、电话来查找记录，并逐项进行修改。

（6）可根据姓名、电话来查找记录，并判断有无满足条件的记录。

（7）排序功能，可以根据姓名进行排序。

并自行添加了如下功能：

（1）保存功能，将数据存储在save.sav文件中

（2）额外对较低版本C标准环境下的成功运行

由于使用了中文字符，请自行调整编码（本文件采用UTF-8编码），需要GBK编码支持

本程序在以下环境成功编译并运行：

Microsoft Visual Studio 2015 + Windows 10

Sublime Text 3 + MinGW + Window 10

程序仅供于参考，不建议抄袭。

已上传GitHub以证明版权所有：

<https://github.com/OhYee/ACM.github.io/blob/master/C%E4%BD%9C%E4%B8%9A/Contacts.c>

**功能实现**：

程序主体采用while循环保证能够不断运行。

通过将相同功能写成函数重复调用，简化代码。

并且充分考虑低版本编译器上的运行情况(仍然建议采用较新的编译器)

数据采用自定义的数据结构存储。

程序的执行上采用了简单的排序算法(冒泡排序)，并进行了简单的优化。

额外添加的记录保存的功能，使程序更具有实用价值。

本程序共涉及到以下知识：

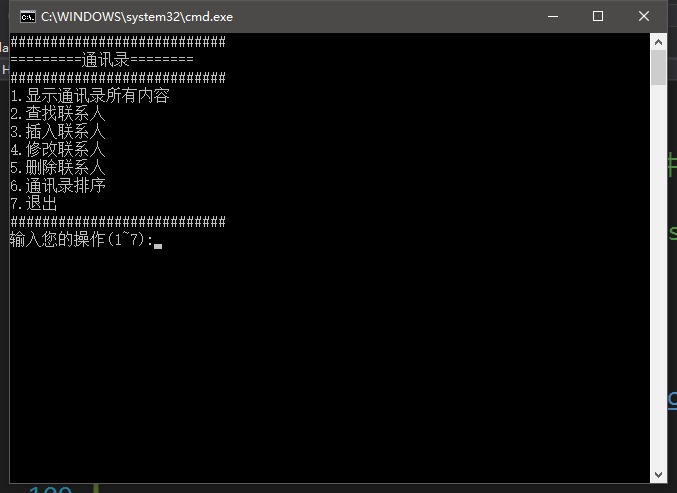
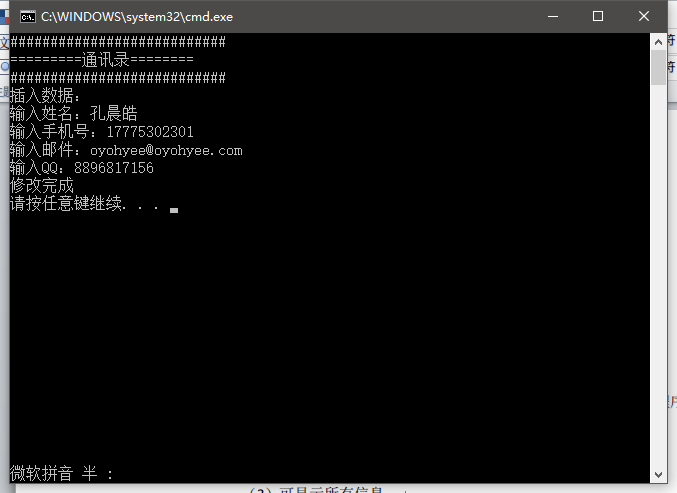
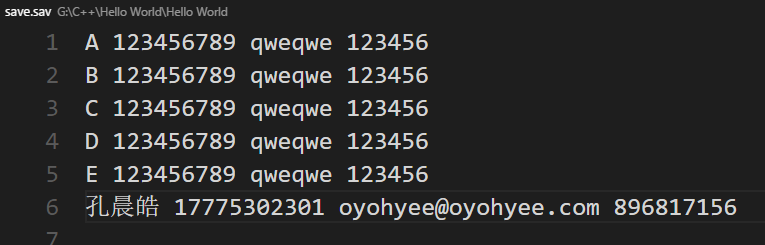
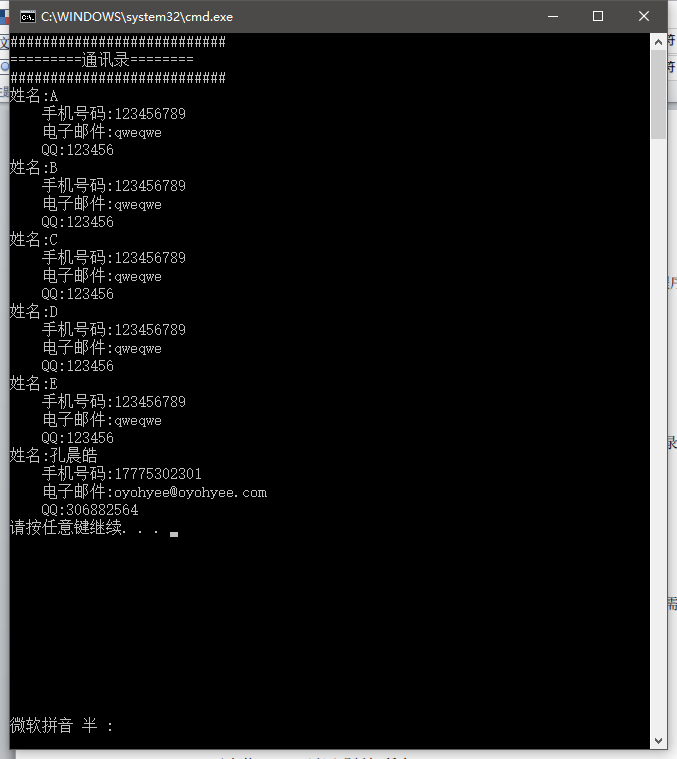
1. 数据的输入和输出
2. 循环、判断
3. 函数的声明及调用
4. 数组的基本操作
5. 宏定义
6. 全局变量和局部变量
7. 结构体以及结构体函数
8. 函数传参的引用
9. 文件指针及读取和写入文件
10. 字符串的操作
11. 排序算法(冒泡排序)

**收获**：

通过编写该程序，本小组成员体会到了编程的实际用途。通过自己的劳动写出了能够帮助自己生活工作的工具，将所学知识进行了实际运用，极大激发了我们学习编程的兴趣。

通过自己动手编写程序，我们了解了如何完整写出一个程序的步骤，培养了我们宏观规划及细节处理的能力。小组成员通过共同工作，学会了团队合作的能力。这是一个团队的根本，有助于我们以后的工作学习。

**运行截图**：

**实现代码**：



#include <stdio.h>

#include <stdlib.h>

#include <string.h>

/\*==============

宏定义部分

==============\*/

//兼容较低版本C标准

#define bool int

#define true 1

#define false 0

//各项系数

#define MAXSIZE 30

#define SIZE\_NAME 20

#define SIZE\_PHONE 15

#define SIZE\_EMAIL 30

#define LINE (printf("###########################\n"))

//兼容低版本C标准并简化循环语句

#define REP(n) for(o=0;o<n;o++)

//比较大小

#define max(a,b) ((a)>(b)?(a):(b))

/\*==============

结构体定义

==============\*/

typedef struct {

char name[SIZE\_NAME];

char phone[SIZE\_PHONE];

char email[SIZE\_EMAIL];

int QQ;

void print() {

printf("姓名:%s\n",name);

printf(" 手机号码:%s\n",phone);

printf(" 电子邮件:%s\n",email);

printf(" QQ:%d\n",QQ);

}

void insert(char a[SIZE\_NAME],char b[SIZE\_PHONE],char c[SIZE\_EMAIL],int d) {

strcpy(name,a);

strcpy(phone,b);

strcpy(email,c);

QQ = d;

}

}People;

/\*==============

全局变量定义部分

==============\*/

People p[MAXSIZE];

int num = 0;

int o,i;//循环用，兼容低版本C标准

/\*==============

函数声明部分

==============\*/

void Do();

void head();

bool menu();

void Read();

void Save();

void Show();

void Search();

void Insert();

void Change();

void Delete();

void Sort();

bool Compare(char a[SIZE\_NAME],char b[SIZE\_NAME]);

void Swap(People &a,People &b);

int Find(char id[],int type);

/\*==============

函数定义部分

==============\*/

//界面\_头部部分

inline void head() {

system("cls");

LINE;

printf("=========通讯录========\n");

LINE;

}

//主菜单

bool menu() {

head();

printf("1.显示通讯录所有内容\n");

printf("2.查找联系人\n");

printf("3.插入联系人\n");

printf("4.修改联系人\n");

printf("5.删除联系人\n");

printf("6.通讯录排序\n");

printf("7.退出\n");

LINE;

int com = 0;

while(!(com >= 1 && com <= 7)) {

printf("输入您的操作(1~7):");

scanf("%d",&com);

}

switch(com) {

case 1:Show();break;

case 2:Search();break;

case 3:Insert();break;

case 4:Change();break;

case 5:Delete();break;

case 6:Sort();break;

case 7:return false;

}

system("pause");

return true;

}

void Show() {

head();

REP(num)

p[o].print();

}

void Search() {

head();

int com = 0;

while(!(com >= 1 && com <= 2)) {

printf("使用以下为索引值搜索(1.姓名 2.手机 -1.返回主界面):");

scanf("%d",&com);

if(com == -1)

return;

}

char f[max(SIZE\_NAME,SIZE\_PHONE)];

printf("输入要查找的值:");

scanf("%s",f);

int it = Find(f,com);

if(it == -1) {

printf("未找到该数据\n");

} else {

printf("找到数据，索引值为:%d\n",o);

p[o].print();

}

}

void Insert() {

head();

if(num == MAXSIZE) {

printf("内存已满\n");

} else {

char name[SIZE\_NAME];

char phone[SIZE\_PHONE];

char email[SIZE\_EMAIL];

int QQ;

printf("插入数据：\n");

printf("输入姓名：");

scanf("%s",name);

printf("输入手机号：");

scanf("%s",phone);

printf("输入邮件：");

scanf("%s",email);

printf("输入QQ：");

scanf("%d",&QQ);

p[num++].insert(name,phone,email,QQ);

printf("修改完成\n");

}

}

void Change() {

head();

printf("改变已有数据：\n");

int index = -2;

while(!(index >= 0 && index < num)) {

printf("请输入索引值(可通过查找功能获取索引值，输入-1返回主界面):\n");

scanf("%d",&index);

if(index == -1)

return;

}

printf("输入姓名：");

scanf("%s",p[index].name);

printf("输入手机号：");

scanf("%s",p[index].phone);

printf("输入邮件：");

scanf("%s",p[index].email);

printf("输入QQ：");

scanf("%d",&p[index].QQ);

printf("修改完成\n");

}

void Delete() {

head();

printf("删除已有数据：\n");

int index = -2;

while(!(index >= 0 && index < num)) {

printf("请输入索引值(可通过查找功能获取索引值，输入-1返回主界面):\n");

scanf("%d",&index);

if(index == -1)

return;

}

for(i = index;i < num;i++)

memcpy(&p[i],&p[i + 1],sizeof(p[i + 1]));

num--;

printf("删除完成\n");

}

void Sort() {

//使用冒泡排序进行排序

int j,k;

bool flag;

k = num;

flag = true;

while(flag) {

flag = false;

for(j = 1; j < k; j++)

if(Compare(p[j - 1].name,p[j].name)) {

Swap(p[j - 1],p[j]);

flag = true;

}

k--;

}

printf("排序完成\n");

}

//从文件读取数据

void Read() {

FILE \* f;

f = fopen("save.sav","r");

if(f) {

char name[SIZE\_NAME];

char phone[SIZE\_PHONE];

char email[SIZE\_EMAIL];

int QQ;

while(fscanf(f,"%s%s%s%d",name,phone,email,&QQ) != EOF)

p[num++].insert(name,phone,email,QQ);

fclose(f);

}

}

//写入数据到文件

void Save() {

FILE \* f;

f = fopen("save.sav","w");

REP(num)

fprintf(f,"%s %s %s %d\n",p[o].name,p[o].phone,p[o].email,p[o].QQ);

fclose(f);

}

//比较两个字符串的大小，按照字典序比较，若a>b返回true

bool Compare(char a[SIZE\_NAME],char b[SIZE\_NAME]) {

int it1 = strlen(a) - 1;

int it2 = strlen(b) - 1;

while(1) {

if(a[it1] != b[it2]) {

if(a[it1] > b[it2])

return true;

else

return false;

}

it1--;it2--;

if(it1 == -1 || it2 == -1)

return it1 < it2;

}

}

//交换

void Swap(People &a,People &b) {

People temp;

memcpy(&temp,&a,sizeof(a));

memcpy(&a,&b,sizeof(b));

memcpy(&b,&temp,sizeof(temp));

}

//查找指定特征值的记录，type 1 按照姓名查找 2 按照手机查找 返回数组下标

int Find(char id[],int type) {

int it = -1;

if(type == 1)

REP(num)

if(strcmp(id,p[o].name) == 0) {

it = o;

break;

} else

REP(num)

if(strcmp(id,p[o].phone) == 0) {

it = o;

break;

}

return it;

}

//循环主体

void Do() {

Read();

while(menu());

Save();

}

//程序入口

int main() {

Do();

return 0;

}