C语言实现通讯录

一、通讯录

实现一个通讯录；

通讯录可以用来存储1000个人的信息，每个人的信息包括：姓名、性别、年龄、电话、住址

提供方法：

- 添加联系人信息
- 删除指定联系人信息
- 查找指定联系人信息
- 修改指定联系人信息
- 显示所有联系人信息
- 清空所有联系人
- 以名字排序所有联系人

注：这是一个简单的通讯录，实现方案是初级版。

只能在程序运行期间存在（没有写入文件）。

二、菜单实现和用户交互

```
void menu()
{
	printf("======================\n");
	printf(" 1. 新增联系人\n");
	printf(" 2. 删除联系人\n");
	printf(" 3. 查找联系人\n");
	printf(" 4. 修改联系人\n");
	printf(" 5. 查看所有联系人\n");
	printf(" 6. 清空所有联系人\n");
	printf(" 7. 以名字排序所有联系人\n");
	printf(" 0. 退出\n");
	printf("======================\n");
}
```

三、主函数

1.enum选项

```
enum Option
{
	EXIT,
	ADD,
	DEL,
	SEARCH,
	MODIFY,
	SHOW,
	EMPTY,
	SORT,
};
```

`enum` 枚举常量，里面默认对应的值是从0~7，刚好和菜单中的选项匹配起来了

2.switch判断

```
do
	{
		menu();
		printf(”请选择：>”);
		scanf(”%d”, &input);
		switch(input)
		{
			case ADD:
				AddContact(&con);
				break;
			case DEL:
				DelContact(&con);
				break;
			case SEARCH:
				SearchContact(&con);
				break;
			case MODIFY:
				ModifyContact(&con);
					break;
			case SHOW:
				ShowContact(&con);
				break;
			case EMPTY:
				EmptyContact(&con);
				break;
			case EXIT:
				printf(”退出通讯录\n”);
				break;
			default:
```

```
                printf("选择错误\n");
                break;
        }

    } while (input);
```

四、定义联系人和通讯录

1.定义联系人结构体

```
#define NAME_MAX 20
#define SEX_MAX 5
#define TELE_MAX 20
#define ADDR_MAX 30

struct PeoInfo
{
    char name[NAME_MAX];
    int age;
    char sex[SEX_MAX];
    char tele[TELE_MAX];
    char addr[ADDR_MAX];
};
```

注意事项：

1. 我使用了四个 `define` 定义的常量，这样要修改最大值的时候更方便
2. 定义的 `struct` 结构体可以存放：姓名、性别、年龄、电话、住址

2.定义通讯录结构体

```
#define MAX 1000
struct Contact
{
    struct PeoInfo date[MAX];
    int sz;
};
```

注意事项：

1. `struct PeoInfo date[MAX]` 这个数组可以存放 `1000` 人的信息
2. `sz` 表示当前通讯录里面的人数

3.定义结构体变量

```
struct PeoInfo con;
```

五、通讯录初始化

```
#include<string.h>
#include”contact.h”
void Initcontact(struct Contact* pc)
{
	pc->sz = 0;
	memset(pc->date, 0, MAX * sizeof(struct PeoInfo));
}
```

注意事项：

1. 将信息都赋值为0，否则 `struct` 里面都是随机值
2. memset开辟了一块动态内存空间，相关知识请查阅我写的博客：关于memset用法

六、新增联系人

```
void AddContact(struct Contact* pc)
{
	struct PeoInfo tmp = { 0 };
	if (pc->sz == MAX)
	{
		printf(”通讯录已满1000人”);
	}
	else
	{
		printf(”请输入名字：>”);
		scanf(”%s”, tmp.name);
		printf(”请输入年龄：>”);
		scanf(”%d”, tmp.age));
		printf(”请输入性别：>”);
		scanf(”%s”, tmp.sex);
		printf(”请输入电话：>”);
		scanf(”%s”, tmp.tele);
```

```
            printf("请输入地址：>");
            scanf("%s", tmp.addr);
            pc->data[pc->sz] = tmp;
            printf("添加成功！");
            pc->sz++;
        }
}
```

注意事项：

每次添加选择放在下标为 `sz` 的 `data` 数组里面

七、查找联系人

注意事项：

1. 我发现无论是删除联系人、修改联系人、查看所有联系人都需要一个查找所有联系人的动作，因此我们写出一个查找所有联系人的函数

```
int FindContactByName(struct Contact* pc, char name[])
{
    int i = 0;
    for (i = 0; i < pc->sz; i++)
    {
        if (strcmp(pc->data[i].name, name) == 0)
        {
            return i;
        }
    }
    return -1;
}
```

注意事项：

1. 用 `strcmp` 库函数来比较 `data` 数组中的名字与要查找的名字是否相同
2. 相同则返回下标，不同则返回-1

```
void SearchContact(struct Contact* pc)
{
    char name[NAME_MAX] = { 0 };
    printf("请输入要查找人的名字：>");
    scanf("%s", name);
    int pos= FindContactByName(pc, name);
    if (-1 == pos)
    {
        printf("查无此人");
    }
    else
    {
        printf("%15s\t%5s\t%8s\t%15s\t%30s\n\n", "姓名", "年龄", "性别", "电话", "地址");
        printf("%15s\t%5s\t%8s\t%15s\t%30s\n",
```

```
                pc->data[pos].name,
                pc->data[pos].age,
                pc->data[pos].sex,
                pc->data[pos].tele,
                pc->data[pos].addr);
        }
}
```

八、删除联系人

```
void DelContact(struct Contact* pc)
{
        if (pc->sz == 0)
        {
                printf("通讯录为空，无法删除\n");
        }
        char name[NAME_MAX] = { 0 };
        printf("请输入要删除人的名字");
        scanf_s("%s", name);
        int pos=FindContactByName(pc,name);//按照名字去查找，找到了就返回下标，未找到就返回-1
        if (pos == -1)
        {
                printf("指定联系人不存在\n");
        }
        else
        {
                int j = 0;
                for (j = pos; j < pc->sz-1; j++)
                {
                        pc->data[j] = pc->data[j + 1];
                }
                pc->sz--;
                printf("删除成功！\n");
        }

}
```

九、修改联系人

```
void ModifyContact(struct Contact* pc)
{
        char name[NAME_MAX] = { 0 };
        printf("请输入要修改人的名字：>");
```

```
        scanf("%s", name);
        int pos = FindContactByName(pc, name);
        if (-1 == pos)
        {
                printf("要修改的人不存在\n");
        }
        else
        {
                printf("请输入新的名字：>");
                scanf("%s", pc->data[pos].name);//选择放在下标为sz的data里面
                printf("请输入新的年龄：>");
                scanf("%d", &(pc->data[pos].age));
                printf("请输入新的性别：>");
                scanf("%s", pc->data[pos].sex);
                printf("请输入新的电话：>");
                scanf("%s", pc->data[pos].tele);
                printf("请输入新的地址：>");
                scanf("%s", pc->data[pos].addr);
        }
}
```

十、查看所有联系人

```
void ShowContact(struct Contact* pc)
{
        int i = 0;
        printf("%15s\t%5s\t%8s\t%15s\t%30s\n\n", "姓名", "年龄", "性别", "电话", "地址");//打印标题
        for (i = 0; i < pc->sz; i++)
        {
                printf("%15s\t%5s\t%8s\t%15s\t%30s\n",
                        pc->data[i].name,
                        pc->data[i].age,
                        pc->data[i].sex,
                        pc->data[i].tele,
                        pc->data[i].addr);
        }
}
```

十一、清空所有联系人

```
void EmptyContact(struct Contact* pc)
{
        pc->sz = 0;
```

```
	memset(pc->data, 0, MAX * sizeof(struct PeoInfo));
}
```

注意事项：

`memset` 不是开辟内存的 是将内存空间中的数据清零的

十二、以名字排序所有联系人

```
void SortContact(struct Contact* pc)
{
	qsort(pc->data, pc->sz, sizeof(struct PeoInfo), CmpByname);
}
```

注意事项：

有关qsort相关的知识，请参考此篇博客

十三、完整代码

contact.h

```
#define _CRT_SECURE_NO_WARNINGS 1
#define _CRT_SECURE_NO_WARNINGS 1
#include<stdio.h>
#define NAME_MAX 20
#define SEX_MAX 5
#define TELE_MAX 20
#define ADDR_MAX 30
#define MAX 100
#include <string.h>
#include <stdio.h>
#include <stdlib.h>

struct PeoInfo
{
	char name[NAME_MAX];
	int age;
	char sex[SEX_MAX];
	char tele[TELE_MAX];
```

```
	char addr[ADDR_MAX];
};
struct Contact
{
	struct PeoInfo data[MAX];
	int sz;
};
//初始化通讯录
void InitContact(struct Contact* pc);

//清空所有联系人
void EmptyContact(struct Contact* pc);

//增加联系人
void AddContact(struct Contact* pc);

//显示所有的联系人
void ShowContact(struct Contact* pc);

//删除指定联系人
void DelContact(struct Contact* pc);

//查找指定联系人
void SearchContact(const struct Contact* pc);

//修改指定联系人
void ModifyContact(struct Contact* pc);
```

test.c

```
#define _CRT_SECURE_NO_WARNINGS 1
#define _CRT_SECURE_NO_WARNINGS 1
#include<stdio.h>
#include ”contact.h”
void menu()
{

	printf(”=====================\n”);
	printf(” 1. 新增联系人\n”);
	printf(” 2. 删除联系人\n”);
	printf(” 3. 查找联系人\n”);
	printf(” 4. 修改联系人\n”);
	printf(” 5. 查看所有联系人\n”);
	printf(” 6. 清空所有联系人\n”);
	printf(” 7. 以名字排序所有联系人\n”);
	printf(” 0. 退出\n”);
	printf(”=====================\n”);
	printf(” 请输入您的选择:”);

}
enum Option
{
	EXIT,
	ADD,
	DEL,
```

```
	SEARCH,
	MODIFY,
	SHOW,
	EMPTY,
	SORT,
};//枚举常量，里面对应的值是从0~6，刚好和菜单匹配起来了
int main()
{
	int input = 0;
	struct Contact con;
	InitContact(&con);
	do
	{
		menu();
		printf("请选择：>");
		scanf_s("%d", &input);
		switch (input)
		{
		case ADD:
			AddContact(&con);
			break;
		case DEL:
			DelContact(&con);
			break;
		case SEARCH:
			SearchContact(&con);
			break;
		case MODIFY:
			ModifyContact(&con);
			break;
		case SHOW:
			ShowContact(&con);
			break;
		case EMPTY:
			EmptyContact(&con);
			break;
		case EXIT:
			printf("退出通讯录\n");
			break;
		default:
			printf("选择错误\n");
			break;
		}

	} while (input);

	return 0;
}
```

contact.c

```
#define _CRT_SECURE_NO_WARNINGS 1
#include<string.h>
#include "contact.h"
void InitContact(struct Contact* pc)
```

```
{
	pc->sz = 0;
	memset(pc->data, 0, MAX * sizeof(struct PeoInfo));
}
void AddContact(struct Contact* pc)
{
	struct PeoInfo tmp = { 0 };
	if (pc->sz == MAX)
	{
		printf("通讯录已满1000人");
	}
	else
	{
		printf("请输入名字：>");
		scanf("%s", tmp.name);//选择放在下标为sz的data里面
		printf("请输入年龄：>");
		scanf("%d", &(tmp.age));
		printf("请输入性别：>");
		scanf("%s", tmp.sex);
		printf("请输入电话：>");
		scanf("%s", tmp.tele);
		printf("请输入地址：>");
		scanf("%s", tmp.addr);
		pc->data[pc->sz] = tmp;
		printf("添加成功！");
		pc->sz++;
	}
}
void ShowContact(struct Contact* pc)
{
	int i = 0;
	printf("%15s\t%5s\t%8s\t%15s\t%30s\n\n", "姓名", "年龄", "性别", "电话", "地址");//打印标题
	for (i = 0; i < pc->sz; i++)
	{
		printf("%15s\t%5d\t%8s\t%15s\t%30s\n",
			pc->data[i].name,
			pc->data[i].age,
			pc->data[i].sex,
			pc->data[i].tele,
			pc->data[i].addr);
	}
}
int FindContactByName(struct Contact* pc, char name[])
{
	int i = 0;
	for (i = 0; i < pc->sz; i++)
	{
		if (strcmp(pc->data[i].name, name) == 0)
		{
			return i;
		}
	}
	return -1;
}

void DelContact(struct Contact* pc)
{
	if (pc->sz == 0)
```

```
	{
		printf("通讯录为空，无法删除\n");
		return;
	}
	char name[NAME_MAX] = { 0 };
	printf("请输入要删除人的名字：>");
	scanf("%s", name);
	//查找
	int pos = FindContactByName(pc, name);
	if (pos == -1)
	{
		printf("指定的联系人不存在\n");
	}
	else
	{
		//删除
		int j = 0;
		for (j = pos; j < pc->sz - 1; j++)
		{
			pc->data[j] = pc->data[j + 1];
		}
		pc->sz--;
		//
		printf("删除成功\n");
	}
}
void SearchContact(struct Contact* pc)
{
	char name[NAME_MAX] = { 0 };
	printf("请输入要查找人的名字：>");
	scanf("%s", name);
	int pos = FindContactByName(pc, name);
	if (-1 == pos)
	{
		printf("查无此人");
	}
	else
	{
		printf("%15s\t%5s\t%8s\t%15s\t%30s\n\n", "姓名", "年龄", "性别", "电话", "地址");
		printf("%15s\t%5d\t%8s\t%15s\t%30s\n",
			pc->data[pos].name,
			pc->data[pos].age,
			pc->data[pos].sex,
			pc->data[pos].tele,
			pc->data[pos].addr);
	}
}
void ModifyContact(struct Contact* pc)
{
	char name[NAME_MAX] = { 0 };
	printf("请输入要修改人的名字：>");
	scanf("%s", name);
	int pos = FindContactByName(pc, name);
	if (-1 == pos)
	{
		printf("要修改的人不存在\n");
	}
	else
```

```
	{
		printf("请输入新的名字：>");
		scanf("%s", pc->data[pos].name);//选择放在下标为sz的data里面
		printf("请输入新的年龄：>");
		scanf("%d", &(pc->data[pos].age));
		printf("请输入新的性别：>");
		scanf("%s", pc->data[pos].sex);
		printf("请输入新的电话：>");
		scanf("%s", pc->data[pos].tele);
		printf("请输入新的地址：>");
		scanf("%s", pc->data[pos].addr);
	}
}
void EmptyContact(struct Contact* pc)
{
	pc->sz = 0;
	memset(pc->data, 0, MAX * sizeof(struct PeoInfo));
}
```