

## 实例 2：手机通讯录

通讯录是记录了联系人姓名和联系方式的名录，手机通讯录是最常见的通讯录之一，人们可以在通讯录中通过姓名查看相关联系人的联系方式、邮箱、地址等信息，也可以在其中新增联系人，或修改、删除联系人信息。下面是一个常见通讯录的功能菜单，如图 1 所示。

```
=====
欢迎使用通讯录：
1. 添加联系人
2. 查看通讯录
3. 删除联系人
4. 修改联系人
5. 查找联系人
6. 退出
=====
```

图 1 通讯录功能菜单

图 1 中的通讯录中包含 6 个功能，每个功能都对应一个序号，用户可根据提示“请输入功能序号”选择序号执行相应的操作，包括：

- (1) 添加联系人：用户根据提示“请输入联系人的姓名：”、“请输入联系人的手机号：”、“请输入联系人的邮箱：”和“请输入联系人的地址：”分别输入联系人的姓名、手机号、邮箱和地址，输入完成后提示“保存成功”。注意，若输入的用户信息为空会提示“请输入正确信息”。
- (2) 查看通讯录：按固定的格式打印通讯录每个联系人的信息。若通讯录中还没有添加过联系人，提示“通讯录无信息”。
- (3) 删除联系人：用户根据提示“请输入要删除的联系人姓名：”输入联系人的姓名，若该联系人存在于通讯录中，则提示“删除成功”，否则提示“该联系人不在通讯录中”。注意，若通讯录中还没有添加过联系人，提示“通讯录无信息”。
- (4) 修改联系人：用户根据提示输入要修改联系人的姓名，之后按照提示“请输入新的姓名：”、“请输入新的手机号：”、“请输入新的邮箱：”、“请输入新的地址：”、分别输入该联系人的新姓名、新手机号、新邮箱、新地址，并打印此时的通讯录信息。注意，若通讯录中还没有添加过联系人，提示“通讯录无信息”。
- (5) 查找联系人：用户根据提示“请输入要查找的联系人姓名”输入联系人的姓名，若该联系人存在于通讯录中，则打印该联系人的所有信息，否则提示“该联系人不在通讯录中”。注意，若通讯录中还没有添加过联系人提示“通讯录无信息”。
- (6) 退出：退出手机通讯录。

本实例要求编写程序，模拟实现如上所述的手机通讯录。

## 实例目标

- 熟练地创建字典
- 掌握字典的基本操作，能添加、修改、删除、查询字典中的元素

## 实例分析

手机通讯录通常包含多个联系人，每个联系人都包含姓名、手机号、邮箱、地址等基本信息，且这些信息之间是相互对应的，因此这里可将联系人视为包含 4 个键值对的字典，将通讯录视为一个包含多个字典的数组，将通讯录中新增联系人、删除联系人、修改联系人、查看联系人的功能视为字典的增删改查操作。

根据以上分析可整理出以下基本实现思路：

- (1) 创建一个空列表，使用该列表存储联系人信息；
- (2) 打印通讯录的功能菜单；
- (3) 创建一个空字典，使用该字典存储联系人的姓名、手机号、邮箱和地址信息。
- (4) 接收用户输入的功能序号，并根据输入的序号执行相应的操作：用户输入“1”执行增加字典元素的操作；用户输入“2”执行查看字典元素的操作；用户输入“3”执行删除字典的操作；用户输入“4”执行修改字典元素的操作；用户输入“5”执行查看字典元素的操作；用户输入“6”执行结束程序的操作。

## 代码实现

```
person_info = []
print("=" * 20)
print('欢迎使用通讯录: ')
print("1.添加联系人")
print("2.查看通讯录")
print("3.删除联系人")
print("4.修改联系人")
print("5.查找联系人")
print("6.退出")
print("=" * 20)
while True:
    per_dict = {}
    fun_num = input('请输入功能序号: ')
    if fun_num == '1':
        per_name = input('请输入联系人的姓名: ')
        phone_num = input('请输入联系人的手机号: ')
```

```
per_email = input('请输入联系人的邮箱: ')
per_address = input('请输入联系人的地址: ')
# 判断输入的是否为空
if per_name.strip() == '' or phone_num.strip() == ''
    or per_email.strip() == '' or per_address.strip() == '':
    print('请输入正确信息')
    continue
else:
    per_dict.update({'姓名': per_name,
                    '手机号': phone_num,
                    '电子邮箱': per_email,
                    '联系地址': per_address})
    person_info.append(per_dict) # 保存到列表中
    print('保存成功')
elif fun_num == '2':
    if len(person_info) == 0:
        print('通讯录无信息')
    for i in person_info:
        print('--*' * 6)
        for title, info in i.items():
            print(title + ':' + info)
        print('--*' * 6)
elif fun_num == '3': # 删除
    if len(person_info) != 0:
        del_name = input('请输入要删除的联系人姓名: ')
        for i in person_info:
            if del_name in i.values():
                person_info.remove(i)
                print(person_info)
                print('删除成功')
            else:
                print('该联系人不在通讯录中')
    else:
        print('通讯录无信息')
elif fun_num == '4': # 修改
    if len(person_info) != 0:
        modi_info = input('请输入要修改的联系人姓名: ')
        for i in person_info:
            if modi_info in i.values():
```

```
        # 获取所在元组在列表中的索引位置
        index_num = person_info.index(i)
        dict_cur_perinfo = person_info[index_num]
        for title, info in dict_cur_perinfo.items():
            print(title + ':' + info)

        modi_name = input('请输入新的姓名: ')
        modi_phone = input('请输入新的手机号: ')
        modi_email = input('请输入新的邮箱: ')
        modi_address = input('请输入新的地址: ')
        dict_cur_perinfo.update(姓名= modi_name)
        dict_cur_perinfo.update(手机号= modi_phone)
        dict_cur_perinfo.update(电子邮箱= modi_email)
        dict_cur_perinfo.update(联系地址= modi_address)
        print(person_info)

    else:
        print('通讯录无信息')

    elif fun_num == '5': # 查找
        if len(person_info) != 0:
            query_name = input('请输入要查找的联系人姓名: ')
            for i in person_info:
                if query_name in i.values():
                    index_num = person_info.index(i)
                    for title, info in person_info[index_num].items():
                        print(title + ':' + info)
                    break
            else:
                print('该联系人不在通讯录中')
        else:
            print('通讯录无信息')

    elif fun_num == '6': # 退出
        break
```

以上代码首先定义了一个空列表 `person_info`，其次打印了通讯录的功能菜单，以提示用户根据序号选择相应的功能，然后创建一个保存联系人的字典 `per_dict`，并接收用户选择的序号 `fun_num`，最后使用 `if-elif-else` 结构处理了不同序号的功能：当 `fun_num` 为“1”时调用 `update()` 方法更新字典元素；当 `fun_num` 为“2”时调用遍历查看字典元素；当 `fun_num` 为“3”时将列表 `person_info` 中相应的字典删除；当 `fun_num` 为“4”时调用 `update()` 方法更新字典元素；当 `fun_num` 为“5”时遍历查看字典元素；当 `fun_num` 为“6”时使用 `break` 语句结束程序。

需要注意的是，为保证程序能一直保持运行，这里需要使用死循环进行控制，由用户执

行退出通讯录的行为。

## 代码测试

运行程序，在控制台输入“1”之后的结果如下所示：

```
=====
欢迎使用通讯录：
1. 添加联系人
2. 查看通讯录
3. 删除联系人
4. 修改联系人
5. 查找联系人
6. 退出
=====
请输入功能序号：1
请输入联系人的姓名：小红
请输入联系人的手机号：123456
请输入联系人的邮箱：123456
请输入联系人的地址：北京
保存成功
```

在控制台输入“2”之后的结果如下所示：

```
请输入功能序号：2
-----*-----*-----*
姓名：小红
手机号：123456
电子邮箱：123456
联系地址：北京
-----*-----*-----*
```

在控制台输入“4”之后的结果如下所示：

```
请输入功能序号：4
请输入要修改的联系人姓名：小红
姓名：小红
手机号：123456
电子邮箱：123456
联系地址：北京
请输入新的姓名：小兰
请输入新的手机号：12345
请输入新的邮箱：12345@163.com
请输入新的地址：北京
```

```
[{'姓名': '小兰', '手机号': '12345', '电子邮箱': '12345@163.com', '联系地址': '北京'}]
```

在控制台输入“5”之后的结果如下所示：

请输入功能序号：5

请输入要查找的联系人姓名：小红

该联系人不在通讯录中

在控制台输入“3”之后的结果如下所示：

请输入功能序号：3

请输入要删除的联系人姓名：小兰

[]

删除成功