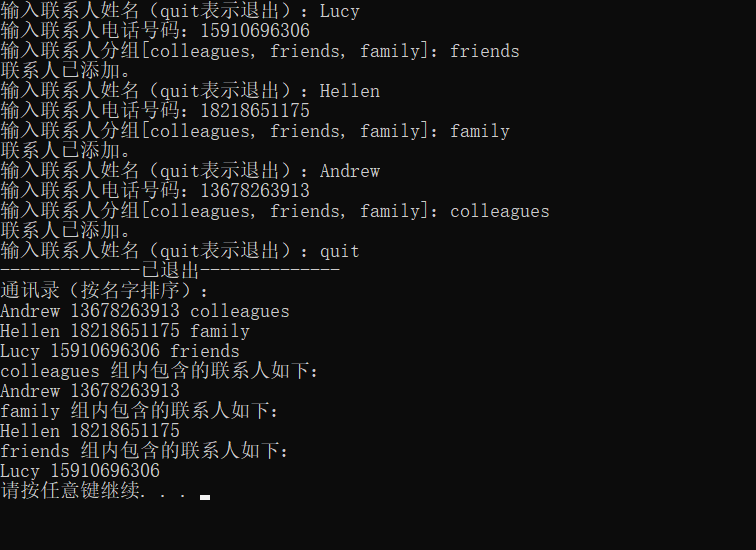
使用java程序设计一个手机通讯录：使用集合类实现一个手机通讯录，可以保存手机联系人的姓名、联系电话、分组信息。主要功能包括：

1. 新增联系人，包括姓名、联系电话和分组信息，不能创建重复名字相同的联系人addContactPerson()
2. 通信录的信息按姓名有序、且姓名不重复printMobileCommunication()
3. 分组打印联系人的信息groupByGroup()、printGroup( )



class ContactPerson implements Comparable<ContactPerson> {

    private String name;

    private String phoneNumber;

    private String group;

    public ContactPerson(String *name*, String *phoneNumber*, String *group*) {

        this.name = *name*;

        this.phoneNumber = *phoneNumber*;

        this.group = *group*;

    }

    public String getName() {

        return name;

    }

    public String getPhoneNumber() {

        return phoneNumber;

    }

    public String getGroup() {

        return group;

    }

    @Override

    public int compareTo(ContactPerson *other*) {

        return name.compareTo(*other*.getName());

    }

}

import java.util.List;

public class CheckUtils {

    public static boolean isValidPhoneNumber(String *phoneNumber*) {

        return *phoneNumber*.matches("0?(13|14|15|18|17)[0-9]{9}");

    }

*// 校验分组信息*

    public static boolean isValidGroup(String *group*, String[] *groups*) {

        for (String validGroup : *groups*) {

            if (*group*.equals(validGroup)) {

                return true;

            }

        }

        return false;

    }

*// 校验姓名是否重复*

    public static boolean isValidName(String *name*, List<String> *nameList*) {

        for (String element : *nameList*) {

            if (*name*.equals(element)) {

                return false;

            }

        }

        return true;

    }

}

import java.util.\*;

public class MobileCommunication {

    private List<ContactPerson> contactList;

    public MobileCommunication() {

        contactList = new ArrayList<>();

    }

    public List<ContactPerson> getContactList() {

        return contactList;

    }

    public void addContactPerson(String *name*, String *phoneNumber*, String *group*) {

        ContactPerson newContact = new ContactPerson(*name*, *phoneNumber*, *group*);

        contactList.add(newContact);

        Collections.sort(contactList);

        System.out.println("联系人已添加。");

    }

    public void printMobileCommunication() {

        System.out.println("通讯录（按名字排序）：");

        for (ContactPerson contact : contactList) {

            System.out.println(contact.getName() + " " + contact.getPhoneNumber() + " " + contact.getGroup());

        }

    }

    public void groupByGroup() {

        Set<String> groups = new HashSet<>();

        for (ContactPerson contact : contactList) {

            groups.add(contact.getGroup());

        }

        for (String group : groups) {

            System.out.println(group + " 组内包含的联系人如下：");

            printGroup(group);

        }

    }

    public void printGroup(String *group*) {

        for (ContactPerson contact : contactList) {

            if (contact.getGroup().equals(*group*)) {

                System.out.println(contact.getName() + " " + contact.getPhoneNumber());

            }

        }

    }

    public List<String> getNameList() {

        List<String> nameList = new ArrayList<>();

        for (ContactPerson contact : contactList) {

            nameList.add(contact.getName());

        }

        return nameList;

    }

    public static void main(String[] *args*) {

        Scanner scanner = new Scanner(System.in);

        MobileCommunication mobileCommunication = new MobileCommunication();

        String[] groupList = { "colleagues", "friends", "family" };

        while (true) {

            System.out.print("输入联系人姓名（quit表示退出）：");

            String name = scanner.nextLine().trim();

*// 非空校验*

            if (name.isEmpty()) {

                System.out.println("联系人姓名不能为空。");

                continue;

            }

*// name合法校验*

            if (!CheckUtils.isValidName(name, mobileCommunication.getNameList())) {

                System.out.println("联系人姓名已存在。");

                continue;

            }

            if (name.equals("quit")) {

                System.out.println("--------------已退出--------------");

                break;

            }

            System.out.print("输入联系人电话号码：");

            String phoneNumber = scanner.nextLine().trim();

*// 校验phoneNumber*

            if (!CheckUtils.isValidPhoneNumber(phoneNumber)) {

                System.out.println("电话号码格式不正确。");

                continue;

            }

            System.out.print("输入联系人分组" + Arrays.toString(groupList) + "：");

            String group = scanner.nextLine().trim();

*// 校验group*

            if (!CheckUtils.isValidGroup(group, groupList)) {

                System.out.println("分组信息不正确。");

                continue;

            }

            mobileCommunication.addContactPerson(name, phoneNumber, group);

        }

        scanner.close();

        mobileCommunication.printMobileCommunication();

        mobileCommunication.groupByGroup();

    }

}