

目录

第一章 需求分析	1
第二章 系统描述	2
第三章 功能模块结构	4
第四章 主要模块算法说明	7
第五章 运行结果	20
第六章 课程设计总结	28
第七章 参考文献	错误! 未定义书签。
附录	错误! 未定义书签。

第一章 需求分析

【问题描述】

设计一个校园导游程序，为来访的客人提供各种信息查询服务。

【基本要求】

- (1)设计每个记录有下列数据项：电话号码、用户名、地址；
- (2)从键盘输入个记录，分别以电话号码和用户名为关键字建立不同散列表存储；
- (3)采用一定的方法解决冲突；
- (4)查找并显示给定电话号码的记录；
- (5)查找并显示给定用户名的记录。

【实现提示】

设计不同的散列函数，尝试不同类型冲突解决方案，考察平均查找长度的变化。

提示：记录与散列表分开，达到不同关键字散列表可共享记录。

【补充内容】

- (1)自动读入硬盘中的记录，并可以选择存储更新后的记录。
- (2)提供信息检测机制，以学号作为唯一关键字，对重复学号的记录不允许插入。
- (3)提供删除功能。
- (4)提供空白检测机制，输入信息任意一项为空则不允许插入。
- (5)提供格式检测机制，输入信息的格式不正确则不允许插入（如年龄不允许输入字符或字符串）
- (6)采用不同的 hash 函数构建方法和不同的冲突处理方式。
- (7)实现用户界面。



第二章 系统描述

1. 开发语言及主要功能实现方法

本程序基于 java 语言写成，配置 java 所需环境变量。

本程序中链表和 hash 函数均未使用 java 库中已有函数，链表和 hash 函数都是使用 java 语言自己编写实现。

Java 语言实现链表和 C 语言类似，但由于 java 没有指针功能，因此可以将节点作为单独的类，用引用的方法实现链式链接。

Hash 函数分别采用除留取余法和伪随机数法，其中伪随机数用于字符串构造 hash 函数，可根据不同的字符串生成不同的随机数。

冲突处理分别采用线性探测法、再哈希法和链地址法。

2. 单条记录所含内容

结合学生电子号码本的需求情况，选出本部周围的 8 项内容，作为一条记录：

学号	姓名	性别	年龄	电话号码	住址	学院	专业
0901150424	王子旭	男	19	18932463803	升华 14-236	信息院	计算机

（其余记录与此格式相同，不做重复）

3. Hash 函数及处理冲突方法

本系统分别对学号、姓名、电话号码三项构建 hash 函数，并采用不同的冲突处



理方法，具体如下：

- (1) 学号散列表：采用除留取余法构建 hash 函数，采用线性探测法处理冲突
- (2) 姓名散列表：采用伪随机数法构建 hash 函数，采用链地址法处理冲突
- (3) 电话号码散列表：采用除留取余法构建 hash 函数，采用再哈希法处理冲突

除留余数法

方法： $f(\text{key}) = \text{key} \bmod p$ ($p \leq m$)， m 是散列表表长

随机数法

方法： $f(\text{key}) = \text{random}(\text{key})$

注意 random 的随机种子需要是固定的，以便查询的时候能够根据 key 重新找到存储位置

适用于关键字长度不等的情况

再哈希法：

方法： $f_i(\text{key}) = RH_i(\text{key})$ ($i=1,2,\dots,k$)

遇到冲突就重新采用一个散列函数计算新的存储位置，可以使关键字不产生聚集

链地址法（拉链）

方法：将所有关键字的同义词记录在一个单链表中，在散列表中只存储所有同义词表的头指针



第三章 功能模块结构

1. 信息插入模块

功能：插入单条记录，依次输入单条记录中的信息

输入要求：学号不允许重复、每项信息不能有空值、性别只允许男 / 女，年龄只允许输入数字等等。

输出结果：将该记录保存在链表中备份，依次采用三种不同的 hash 函数构建散列表索引项，并处理冲突，实现记录与散列表分开，达到不同关键字散列表可共享记录。

2. 信息修改模块

功能：按照输入学号修改相应记录，在屏幕上显示对应记录的所有信息并允许修改。

输入要求：学号不允许重复、每项信息不能有空值、性别只允许男 / 女，年龄只允许输入数字等等。

输出结果：将原记录在链表中的内容更新备份，同时更新三种不同的 hash 函数构建的散列表索引项。

3. 信息删除模块

功能：删除单条记录，同时删除链表中的记录和各散列表中的索引。

输入要求：相应记录的学号必须存在。

输出结果：同时删除链表中的记录和各散列表中的索引。



4. 信息查询模块——按照学号查找

功能：根据学号的散列值查找单条记录，实际上为查找学号散列表中的索引并获取对应的完整记录。

输入要求：相应记录的学号必须存在。

输出结果：在屏幕上显示相关记录。

5. 信息查询模块——按照电话查找

功能：根据电话号码的散列值查找单条记录，实际上为查找电话号码散列表中的索引并获取对应的完整记录。

输入要求：相应记录的电话号码必须存在。

输出结果：在屏幕上显示相关记录。

6. 信息查询模块——按照姓名查找

功能：根据姓名的散列值查找单条记录，实际上为查找姓名散列表中的索引并获取对应的完整记录。

输入要求：相应记录的姓名必须存在。

输出结果：在屏幕上显示相关记录（由于姓名可能重复，所以可能会显示多条记录）。

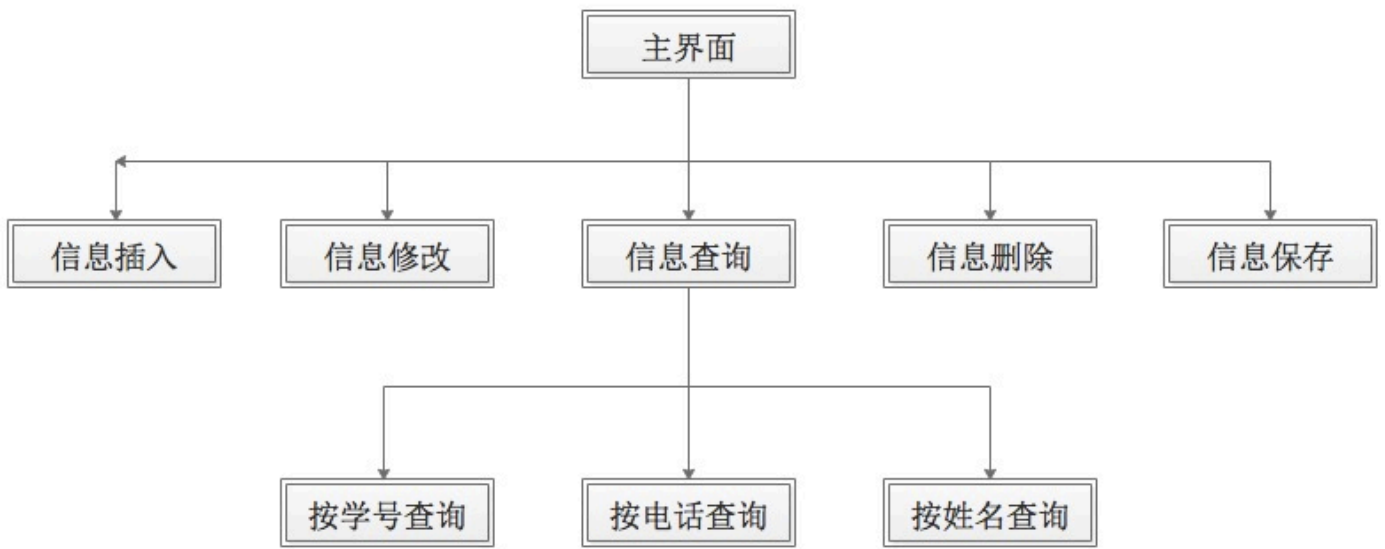
7. 信息保存

功能：将已有记录保存在本地

输入要求：无

输出结果：在本地保存所有信息。





第四章 主要模块算法说明

1. 学号散列表设计:

(1) hash 函数构建方法:

通过除留取余法构建散列表

```
public int hash(int key)
{
    return key % arraySize;
}
```

(2) 处理冲突方法:

采用线性探测法，一次一步

```
while(hashArray[hashVal] != null )
    {
        // 查找下一个位置
        hashVal =(hashVal+1 )%arraySize;
    }
```

(3) 将记录映射到散列表完成步骤:

```
public void insert(Person item)
{
    int key = Integer.parseInt(item.ID);// 获取数据项的关键字，用于计算哈希值
```

```
    int hashVal = hash(key); // 计算哈希值
```

```
    // 当前位置存有数据并且该数据未被删除
```

```
    while(hashArray[hashVal] != null )
    {
        // 查找下一个位置
        hashVal =(hashVal+1 )%arraySize;
    }
```

```
    hashArray[hashVal] = item;    // 找到位置
}
```



(4) 将通过姓名查找：

```
public int find(String key)    // 表中是否存在该关键字的数据项
{
    int hashVal = hash(Integer.parseInt(key));

    while(hashArray[hashVal] != null)
    {
        if(hashArray[hashVal].ID.equals(key))
            return hashVal;
        hashVal=(hashVal + 1) %arraySize;
    }
    return -1;
}

public boolean modify(String key,Person newPerson)
{
    int index = find(key);
    if(index!=-1)
    {
        hashArray[index] = null;
        hashArray[index] = newPerson;
        return true;
    }
    return false;
}
```

(5) 删除散列表中信息

```
public boolean delete(String key) // 根据关键字删除数据
{
    int hashVal = hash(Integer.parseInt(key)); // 根据关键字计算哈希值

    while(hashArray[hashVal] != null) // 该位置存有数据
    {
        // 两者的关键字是否相同
        if(hashArray[hashVal].ID.equals(key))
        {
```



```
        hashArray[hashVal] = null;           // 删除
        return true;
    }:
```

2. 电话号码散列表设计:

(1) hash 函数构建方法:

通过伪随机数法构建散列表

```
public int hash(String key)
{
    return Math.abs(key.hashCode()) % arraySize;
}
```

(2) 处理冲突方法:

采用再哈希法

```
public int rehash(String key) //再哈希
{
    return ((2 * Math.abs(key.hashCode()) + 1) % arraySize);
}
```

(3) 将记录映射到散列表完成步骤:

```
public void insert(Person item)
{
    String key = item.phone; // 获取数据项的关键字，用于计算哈希值

    int hashVal = hash(key); // 计算哈希值

    int stepSize = rehash(key); // 计算步长

    // 当前位置存有数据并且该数据未被删除
```



```
while(hashArray[hashVal] != null )
{
    // 查找下一个位置

    int i = 1;
    hashVal =(hashVal+stepSize)%arraySize;
}

hashArray[hashVal] = item;    // 找到位置
}
```

(4) 将通过电话号码查找:

```
public int find(String key)    // 表中是否存在该关键字的数据项
{
    int hashVal = hash(key);

    int stepSize = rehash(key); // 计算步长

    while(hashArray[hashVal] != null)
    {
        if(hashArray[hashVal].phone.equals(key))
            return hashVal;
        hashVal=(hashVal + stepSize) %arraySize;
    }
    return -1;
}
```

(5) 删除散列表中信息

```
public boolean delete(String key) // 根据关键字删除数据
{
    int hashVal = hash(key); // 根据关键字计算哈希值

    int stepSize = rehash(key); // 计算步长

    while(hashArray[hashVal] != null) // 该位置存有数据
    {
        // 两者的关键字是否相同
        if(hashArray[hashVal].phone.equals(key) )
        {

```



```

        hashArray[hashVal] = null;           // 删除
        return true;
    }

    hashVal=(hashVal + stepSize) %arraySize;    // 关键字不相同，继续
    查找下一个
    }
    return false;                             // 未找到
}

```

3. 姓名散列表设计：

(1) hash 函数构建方法：

通过伪随机数法构建散列表

```

public int hashFunc(String key)           // 计算哈希值
{
    return Math.abs(key.hashCode()% arraySize);
}

```

(2) 处理冲突方法：

采用链地址法（插入、删除、更新、查询全部包含在内）

```

class listNode           //定义节点表
{
    // 可以动态存储数据，扩充容量
    public Person person;

    public listNode next; // 链接到下一个
}

```



```
public ListNode(Person node)                //构造函数
{
    person = node;
}

public String getKey()                       //获取关键字
{
    return person.name;
}

public String getID()                       //获取 ID
{
    return person.ID;
}

}

class Link //LinkedList
{
    public ListNode head;                   // 链表头

    public Link()                          // 构造器
    {
        head = null;
    }

    public void insert(Person person) // 插入
    {
        ListNode node = new ListNode(person);
        node.next = head; //头插
        head = node;
    }

    public boolean delete(String key, String id) //删除
    {
        ListNode previous = null;
        ListNode current = head;

        while( current != null
            && !key.equals(current.getKey()) && !id.equals(current.getID()) ) //未找到
        {
            previous = current;
        }
    }
}
```



```
        current = current.next;    // 查找下一个
    }

    if(previous==null)            // 要删除数据项为表头

        head = head.next;        //      删除表头

    else                            //      不是表头

        previous.next = current.next; //      删除 current

    if(current==null)return false;//删除失败
    else return true;

}

public Person[] find(String key)    // 查找
{
    java.util.ArrayList<Person> arrayList = new java.util.ArrayList<Person>();
    int num = 0;
    listNode current = head;

    while(current != null )
    {
        if(key.equals(current.getKey()))    // 找到
        {
            arrayList.add(current.person);
            num++;
        }
        current = current.next;
    }
    Person p[] = new Person[num];
    for(int i=0;i<num;i++)
    {
        p[i] = arrayList.get(i);
    }
    if(num==0)return null;
    return p;
}
```



```
    }  
}  
  
public class Hash_name  
{  
    private Link[] hashArray;  
    private int arraySize;  
  
    public Hash_name(int size)  
    {  
        arraySize = size;  
  
        hashArray = new Link[arraySize]; // 初始化数组，数组中存储的是链表  
  
        for(int j=0; j<arraySize; j++) // 初始化每个数组元素  
            hashArray[j] = new Link();  
    }  
  
    public int hashFunc(String key) // 计算哈希值  
    {  
        return Math.abs(key.hashCode())% arraySize;  
    }  
  
    public void insert(Person person) // 插入数据  
    {  
        String key = person.name; //获取关键字  
  
        int hashVal = hashFunc(key); // 计算关键字哈希值  
  
        hashArray[hashVal].insert(person); // 插入哈希表中对应的位置  
    }  
  
    public boolean delete(String key,String id) // 根据关键字删除数据  
    {
```



```
int hashVal = hashFunc(key);    // 计算关键字哈希值
```

```
boolean flag = hashArray[hashVal].delete(key,id); // 删除哈希表中对应数
```

据

```
return flag;  
}
```

```
public Person[] find(String key)    // 查找
```

```
{  
    int hashVal = hashFunc(key);  
    Person p[] = hashArray[hashVal].find(key);  
    return p;  
}
```

```
public void saved() throws Exception{
```

```
    File file = new File("hash_name.txt");  
    FileOutputStream fos = new FileOutputStream(file);  
    PrintStream ps = new PrintStream(fos);  
    for(int i=0;i<arraySize;i++)  
    {  
        Link node = hashArray[i];  
        if(node==null)  
            continue;  
        listNode target = node.head;  
        while(target!=null)  
        {  
            ps.println("=====");  
            ps.println("学号: "+target.person.ID);  
  
            ps.println("姓名: "+target.person.name);  
  
            ps.println("性别: "+target.person.sex);  
  
            ps.println("年龄: "+target.person.age);  
  
            ps.println("电话: "+target.person.phone);  
  
            ps.println("地址: "+target.person.address);  
  
            ps.println("学院: "+target.person.department);  
        }  
    }  
}
```



```
        ps.println("专业: "+target.person.major);
        ps.println("=====");
        ps.println();
        ps.println();
        target = target.next;
    }
}

ps.close();

}

}
```

4. 记录插入模块设计：

从界面上文本框中读取字符串，并进行判断：首先各项都必须按照要求，比如年龄不能为字符，学号不能为字符等等；然后判断是否有某项为空，有空则不允许插入；最后检测学号是否重复，重复则不允许插入。

然后在链表中添加完整记录，同时建立学号、电话号码、姓名的索引。

```
String ID = jtf1.getText();
String name = jtf2.getText();
String sex = choose;
String age = jtf4.getText();
String phone = jtf5.getText();
String address = jtf6.getText();
String department = jtf7.getText();
String major = jtf8.getText();
```



```

        try{
            Integer.parseInt(ID);
            Integer.parseInt(age);
            Double.parseDouble(phone);

        } catch (Exception ex) {

            JOptionPane.showMessageDialog(this, "学号、年龄或电话号码格式错误");

            return;
        }

        if(ID.equals(null)||name.equals(null)||sex.equals(null)||age.equals(null)||phone.equals(
null)||

            address.equals(null)||department.equals(null)||major.equals(null)

            ||ID.equals("")||name.equals("")||sex.equals("")||age.equals("")||phone.equals("")||
            address.equals("")||department.equals("")||major.equals(""))
        {

            JOptionPane.showMessageDialog(this, "记录不能有空值");

            return;
        }

        int n = JOptionPane.showConfirmDialog(this, "确认添加? ", "确
认",JOptionPane.OK_CANCEL_OPTION);

        if(n!=0)return;
        Person p = new Person(ID, name, sex, age, phone, address,
department, major);
        boolean flag = linklist.addNode(p);
        if(flag==false)
        {

            JOptionPane.showMessageDialog(this, "插入失败，请检查 ID
是否重复");

            return;

```



```
    }  
    hash_ID.insert(p);  
    hash_phone.insert(p);  
    hash_name.insert(p);  
  
    int m = JOptionPane.showConfirmDialog(this, "继续添加? ", "确  
认", JOptionPane.OK_CANCEL_OPTION);  
    if(m!=0)this.dispose();
```

5. 记录修改模块设计:

首先根据给定的学号选择相应记录，显示在屏幕上，允许用户自行修改。

从界面上文本框中读取字符串，并进行判断：首先各项都必须按照要求，比如年龄不能为字符，学号不能为字符等等；然后判断是否有某项为空，有空则不允许插入；最后检测学号是否重复，重复则不允许插入。

然后在链表中更新完整记录，同时更新学号、电话号码、姓名的索引。

算法同上，不再重复。

6. 记录删除模块设计:

删除输入学号对应的记录，同时删除链表中的节点和各索引表中



的索引。

```
String id = JOptionPane.showInputDialog("请输入你要修改的记录的 ID 号：");

    Person target = linklist.searchNode(id);
    if(target==null)
    {
        JOptionPane.showMessageDialog(this, "此 ID 不存在！");

        return;
    }
    linklist.deleteNode(target.ID);
    hash_ID.delete(target.ID);
    hash_phone.delete(target.phone);
    hash_name.delete(target.name,id);

    JOptionPane.showMessageDialog(this, "成功删除！");
```

第五章 运行结果

1.主界面



2.添加记录

添加新记录

学号: 0901150424

姓名: 王子旭

性别: ☒ 男 ☐ 女

年龄: 19

电话: 18932463803

住址: 升华公寓14栋236寝室

学院: 信息科学与工程学院

专业: 计算机科学与技术

确定 取消

出现非法数据拒绝添加记录



3.修改记录



请输入你要修改的记录的ID号:

0901150424

取消 确定

通过学号查找记录并修改

修改记录

学号: 0901150424

姓名: 王子旭

性别: ☒ 男 ☐ 女

年龄: 19

电话: 18932463803

住址: 升华公寓14栋236寝室

学院: 信息科学与工程学院

专业: 计算机科学与技术

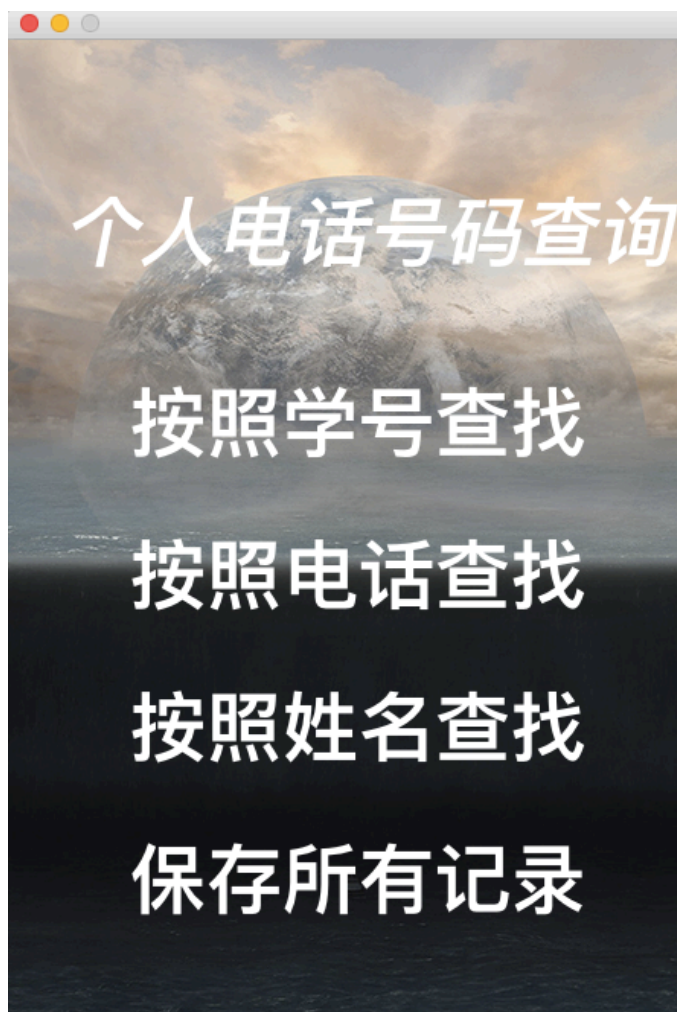
确定

4.删除记录

输入相应学号后，学号存在则删除，不存在则操作失败



5.查询界面



按照学号查找



A screenshot of a macOS-style dialog box titled "输入" (Input). On the left is a logo featuring a stylized flame above three stacked coffee cups. To the right of the logo, the text "请输入你要查询的记录的ID号:" (Please enter the ID number of the record you want to search:) is displayed. Below this text is a text input field containing the value "0901150424". At the bottom right of the dialog are two buttons: "取消" (Cancel) and "确定" (OK).

查找信息如下



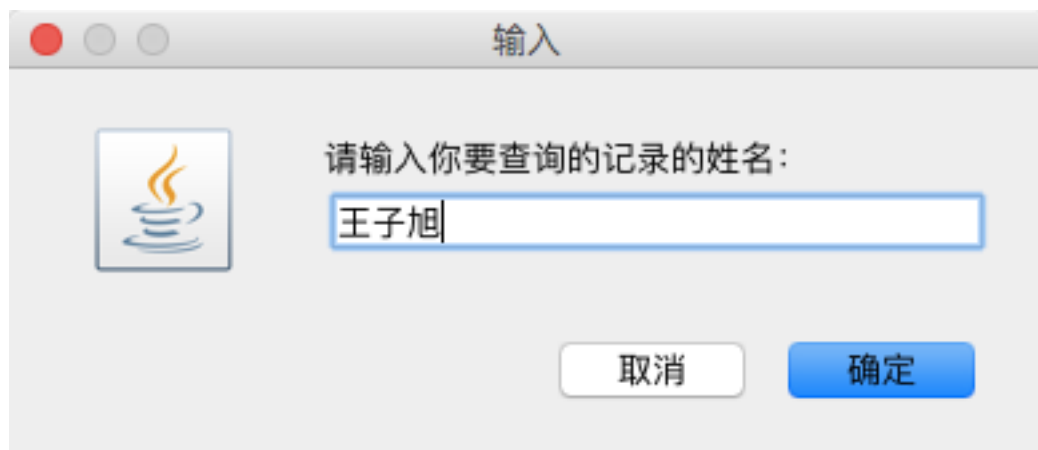
A screenshot of a search results window with a dark, starry background. The title "你要查找的记录如下" (The record you are searching for is as follows) is at the top. Below it, a list of personal information is displayed in a white, monospaced font. The information includes the student ID, name, gender, age, phone number, address, college, and major.

学号:	0901150424
姓名:	王子旭
性别:	男
年龄:	19
电话:	18932463803
住址:	升华公寓14栋236寝室
学院:	信息科学与工程学院
专业:	计算机科学与技术

按照电话号码查找以及按照姓名查找显示同上



A macOS-style dialog box titled "输入" (Input). It features a Java logo icon on the left. The text "请输入你要查询的记录的电话号码:" (Please enter the phone number of the record you want to query:) is displayed. Below it is a text input field containing "18932463803". At the bottom right are two buttons: "取消" (Cancel) and "确定" (OK).



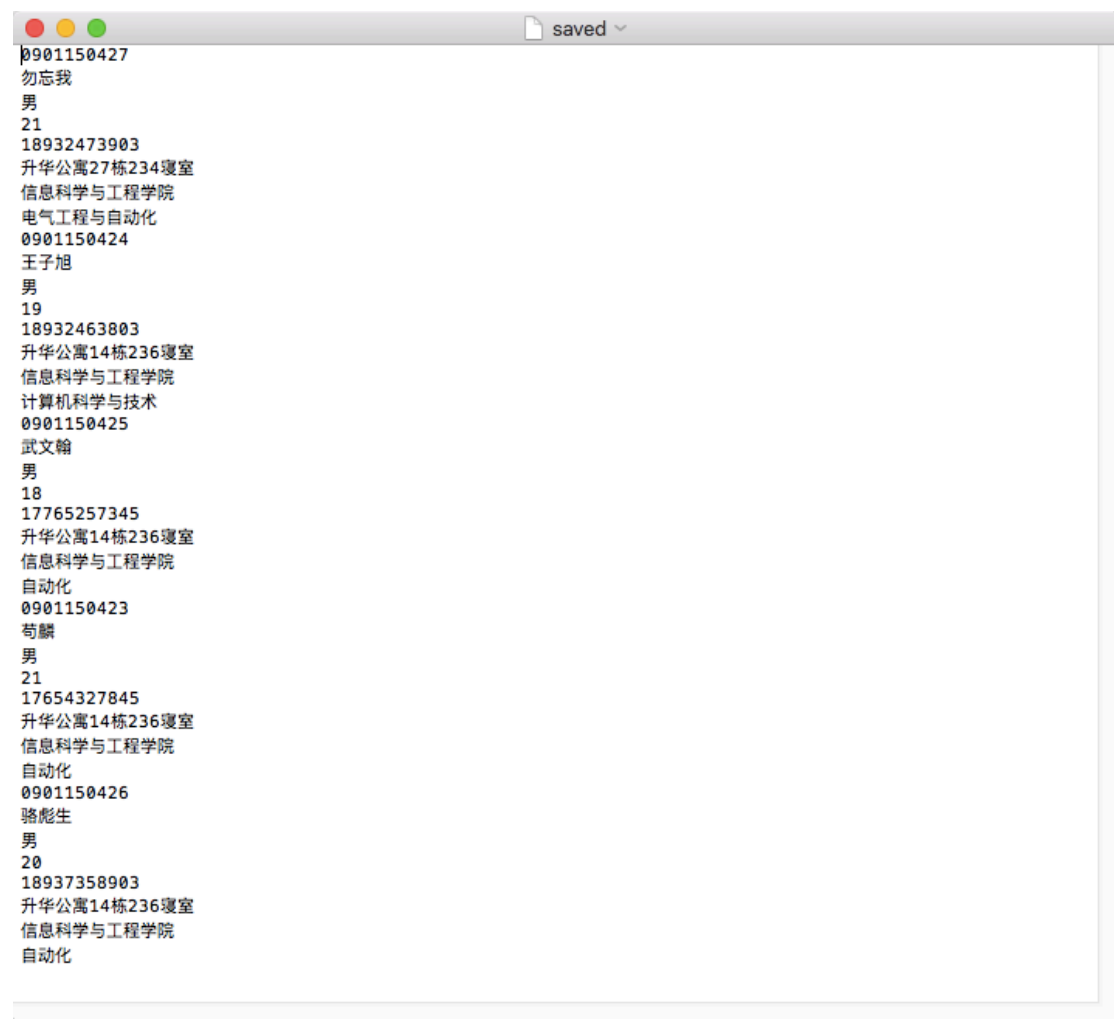
A macOS-style dialog box titled "输入" (Input). It features a Java logo icon on the left. The text "请输入你要查询的记录的姓名:" (Please enter the name of the record you want to query:) is displayed. Below it is a text input field containing "王子旭" (Wang Zixu). At the bottom right are two buttons: "取消" (Cancel) and "确定" (OK).

除此之外，还可以在本地保存所有信息



A macOS-style message dialog box titled "消息" (Message). It features a Java logo icon on the left. The text "已成功保存在本地!" (Successfully saved to local!) is displayed. At the bottom right is a single button labeled "确定" (OK).





为方便调试，在将链表中信息保存下来的同时，也保留了按照不同散列表索引值查找的信息，为方便观察记录在散列表中的存储和次序。

第六章 课程设计总结

问题

1. 人机交互的问题，用户界面如何设计得美观且易于使用，在此使用 java UI 设计，设计友好美观的界面，方便用户使用。
2. 怎样能利用合理的哈希函数构建算法和冲突处理算法。
3. 如何实现记录与散列表分开，达到不同关键字散列表可共享记录。



