苏 州 大 学 实 验 报 告

院、系	计算机学院	年级专业	19 计科图灵	姓名	张昊	学号	1927405160
课程名称	Java 程序设计					成绩	
指导教师	孔芳	同组实验者	无		实验日期	2021 年 4 月 8 日	

实 验 名 称 上机综合 2

一、实验目的

掌握类的基础知识、文件、集合类、基本数据结构的使用。

二、实验内容

给定三个用 UTF-8 编码方式进行编码的文本文件，它们的格式及含义如下：

1）Stddata.txt 存放了若干学生的基本信息，每个学生一行，各基本信息间以空格分隔（格式：14 位学号 姓名 入学年份 专业 联系电话

例如： 20188374858604 宋和科 2018 电器工程 133279242）

2）Recorddata.txt 存放了一卡通的用卡情况，一张卡可以有若干消费记录，一行对应一条消费记录，消费记录的各信息间以空格分隔

（格式：10 位卡号 用卡类型 发生金额 发生地点编号

例如： 0000000007 0 28.72 2）

用卡类型有两种：0-表示存钱操作；1-表示消费操作

发生地点变化目前有 4 个点，编号 0～3

3）StdCard.txt 存放了学生持有的一卡通的状况，学生与一卡通间一一对应，每个学生一行，各基本信息间以空格分隔

（格式：14 位学号 10 位一卡通卡号

例如：20181028947850 0000000005-表示学号是 20181028947850 的学生使用的一卡通卡号为 0000000005）

题目要求：

1) 合理设计：类内部的数据及方法，以及类与类之间的关系。

2) 完成如下一些功能：

（1） 从消费情况文件 Recorddata.txt 文件中加载所有用卡信息，从 Stddata.txt 文件中读取所有学生信息，从学生持卡文件 StdCard.txt 中读取学生持卡状况，完成“学生-一卡通-消费”完整的状态构建。

（2） 统计每位学生一卡通的消费情况（注：存钱类操作不关注），并按消费金额的非降序对学生进行排序，将排序后的学生基本信息、其对应的一卡通基本信息和消费总金额发送到屏幕显示，显示格式如下所示：

```
**************************************************
学号：20171243670365
姓名：黄法框
入学年份:2017
专业：材料化学
联系方式：130156195
卡号：0000000002
余额：182.61
该卡共消费：190.30
```

（3） 将存钱最多的学生的基本信息、其对应的一卡通信息，以及他的详细的用卡信

息以文本文件的形式保存到 SaveTop.txt 文件，文件的基本格式如下所示：

```
**************************************************
学号：20181028945381
姓名：黄费句
入学年份:2018
专业：计算机科学
联系方式：158621799
卡号：0000000004
余额：32.00
该卡的详细消费状况：
存钱 金额：0.11 消费地点：0
存钱 金额：31.43 消费地点：3
存钱 金额：33.65 消费地点：1
存钱 金额：35.96 消费地点：2
消费 金额：22.01 消费地点：0
消费 金额：47.14 消费地点：0
该卡共消费：69.15
```

三、实验过程

设计三个实体类 Student、Card、Record，分别表示学生、一卡通、消费记录，各类的实现如下：

```java
// 学生类
public class Student {
    private final String id;    // 14 位学号
    private final String name;  // 姓名
    private final int year;     // 入学年份
    private final String major; // 专业
    private final String phone; // 联系电话
    public Student(String id, String name, int year,
                   String major, String phone) {
        this.id = id;       this.name = name;
        this.year = year;   this.major = major;
        this.phone = phone;
    }
    public String getId() { return this.id; }
    public String getName() { return this.name; }
    public int getYear() { return this.year; }
    public String getMajor() { return this.major; }
    public String getPhone() { return this.phone; }
    @Override public String toString() {
        return "学号：" + this.id + System.lineSeparator() +
               "姓名：" + this.name + System.lineSeparator() +
               "入学年份：" + this.year + System.lineSeparator() +
               "专业：" + this.major + System.lineSeparator() +
               "联系方式：" + this.phone;
    }
}
```

```java
// 一卡通类
public class Card {
    private final String cardId;       // 10 位一卡通卡号
    private final List<Record> records; // 一卡通的用卡情况
    private double balance;            // 余额
    private double consume;            // 消费总金额
    private double deposit;            // 存款总金额
    public Card(String id) {
        this.cardId = id;      this.records = new LinkedList<>();
        this.balance = 0.0;    this.consume = 0.0;
        this.deposit = 0.0;
    }
    /**
     * 新增一条用卡信息
     * @param record 用卡信息（消费记录类实例）
     */
    public void insert(Record record) {
        this.records.add(record);
        switch (record.getType()) {
            case DEPOSIT:
                this.balance += record.getAmount();
                this.deposit += record.getAmount();   break;
            case CONSUME:
                this.balance -= record.getAmount();
                this.consume += record.getAmount();  break;
        }
    }
    public String getCardId() { return this.cardId; }
    public double getConsume() { return this.consume; }
    public double getDeposit() { return this.deposit; }
    public double getBalance() { return this.balance; }
    // 简明信息
    public String briefInformation() {
        return "卡号: " + this.cardId + System.lineSeparator() +
               String.format("余额: %.2f", this.balance) +
               System.lineSeparator() +
               String.format("该卡共消费: %.2f", this.consume);
    }
    // 详细信息
    public String detailedInformation() {
        StringBuilder str = new StringBuilder();
        str.append("卡号：").append(this.cardId)
            .append(System.lineSeparator())
            .append(String.format("余额：%.2f", this.balance))
```

```
            .append(System.lineSeparator());
        for (Record record : this.records) {
            str.append(record.consumptionInformation())
                .append(System.lineSeparator());
        }
        str.append(String.format("该卡共消费：%.2f", this.consume));
        return str.toString();
    }
}
// 消费记录类实例
public class Record {
    // 用卡类型枚举
    public enum RecordType {
        DEPOSIT,    // 存钱操作
        CONSUME     // 消费操作
    }
    private final String cardId;        // 卡号
    private final RecordType type;      // 用卡类型
    private final double amount;        // 发生金额
    private final int place;           // 发生地点编号
    public Record(String cardId, RecordType type,
                  double amount, int place) {
        this.cardId = cardId;       this.type = type;
        this.amount = amount;       this.place = place;
    }
    public String getCardId() { return cardId; }
    public RecordType getType() { return this.type; }
    public double getAmount() { return this.amount; }
    public int getPlace() { return this.place; }
    // 消费明细
    public String consumptionInformation() {
        String typeString = "";
        switch (this.type) {
            case DEPOSIT: typeString = "存钱"; break;
            case CONSUME: typeString = "消费"; break;
        }
        return String.format("%s 金额：%.2f 消费地点：%d",
                             typeString, this.amount, this.place);
    }
}
```

其中，Card 类持有 Record 实例的集合，Record 类中使用枚举类型 RecordType 标识存钱、消费动作。

设计了一个状态管理类 StudentCardManager，用于构建“学生-一卡通-消费”持有关系。类中使用 ArrayList 来保存所有学生信息，通过 HashMap 来实现从一卡通卡号到一卡通实例

的快速查找，使用 HashMap 保存从学生卡号到一卡通卡号的映射关系。一卡通内的 record 字段保存了消费信息。并且在 StudentCardManager 类中提供添加学生的 addStudent 方法，关联学生与一卡通的 linkStudentCard 方法，通过一卡通卡号找一卡通实例的 findCardByCardId 方法，通过学生学号查找一卡通实例的 findCardByStudentId 方法，以及添加消费记录的 addRecord 方法。具体实现如下：

```
public class StudentCardManager {
    private final List<Student> students;
    private final Map<String, Card> cards;
    private final Map<String, String> studentCardMap;
    public StudentCardManager() {
        this.students = new ArrayList<>();
        this.cards = new HashMap<>();
        this.studentCardMap = new HashMap<>();
    }
    public void addStudent(String id, String name, int year,
                           String major, String phone) {
        this.students.add(new Student(id, name, year, major, phone));
    }
    public void linkStudentCard(String studentId, String cardId) {
        this.cards.put(cardId, new Card(cardId));
        this.studentCardMap.put(studentId, cardId);
    }
    public Card findCardByCardId(String id) {
        return this.cards.getOrDefault(id, null);
    }
    public Card findCardByStudentId(String id) {
        if (!this.studentCardMap.containsKey(id))
            return null;
        return this.findCardByCardId(this.studentCardMap.get(id));
    }
    public void addRecord(String cardId, Record.RecordType type,
                          double amount, int place) {
        Card card = this.findCardByCardId(cardId);
        if (card == null)
          throw new NoSuchElementException("未能找到卡号为"+cardId+"的一卡通");
        card.insert(new Record(cardId, type, amount, place));
    }
    public List<Student> getStudents() { return students; }
}
```

设计了一个文件处理类，基于 BufferedReader 和 BufferedWriter 分别实现了文件的读取和写入，并定义枚举类 IOMode 以标识文件打开方式。在文件读取方法中，使用 Java8 stream 以及 Consumer 接口引用实现了一个通用的文件读取方法，调用时传入实现了 Consumer 接口的对象，该对象具体定义了文件的读取方法。在文件写入方法中，使用不定长参数，将传入的多个字符串逐行写入文件。具体实现如下：

```
public class FileManager {
    public enum IOMode { READ, WRITE }
    public static final IOMode defaultIOMode = IOMode.READ;
    private String filePath;
    private IOMode mode;
    public FileManager(String filePath, IOMode mode) {
        this.open(filePath, mode);
    }
    public FileManager(String filePath) {
        this.open(filePath);
    }
    public FileManager() {
        this(null, defaultIOMode);
    }
    public void open(String filePath, IOMode mode) {
        this.filePath = filePath;       this.mode = mode;
    }
    public void open(String filePath) {
        this.open(filePath, defaultIOMode);
    }
    public void setMode(IOMode mode) { this.mode = mode; }
    /**
     * 文件读取方法，从指定文件中一行一行读取
     * 使用接口 WriteLineHandler 中的 handle 方法处理各行
     * 仅支持打开类型为 IOMode.READ 的对象调用
     * @param action 处理各行的动作
     */
    public void readLines(Consumer<String> action)
                    throws IOException {
        if (this.filePath == null || this.mode != IOMode.READ) {
            throw new IOException("读文件被拒绝，FILE: " +
                        this.filePath + "; MODE: " + this.mode);
        }
        try (FileInputStream in = new FileInputStream(this.filePath);
  BufferedReader reader = new BufferedReader(new InputStreamReader(in))) {
            // try-with-resources 结构管理文件对象
            reader.lines().forEachOrdered(action);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
    /**
     * 保存对象至目标文件（调用 toString 方法）
     * 仅支持打开类型为 IOMode.WRITE 的对象调用
```

```
     * @param strings (不定长参数) 要保存的（多个）对象
     */
    public void writeLines(String... strings) throws IOException {
        if (this.filePath == null || mode != IOMode.WRITE) {
            throw new IOException("写文件被拒绝，FILE: " +
this.filePath + "; MODE: " + this.mode);
        }
        try (FileWriter file = new FileWriter(this.filePath);
             BufferedWriter writer = new BufferedWriter(file)) {
             // try-with-resources 结构管理文件对象
            for (String str : strings) { // 将 strings 逐行写入文本文件
                writer.write(str);
                writer.newLine();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

设计了一个主类 Execute，在主类中完成“学生-一卡通-消费”完整的状态构建。主类中包含状态管理类 StudentCardManager 实例，并定义如下主函数中调用的方法供状态构建与任务执行：

```
public static void main(String[] args) {
    Execute execute = new Execute();
    // 读取文件，构建 "学生-一卡通-消费" 状态
    execute.establishState();
    // 统计每位学生一卡通的消费情况
    execute.statisticsAndSortStudent();
    // 找到存钱最多的学生，将其基本信息、其对应的一卡通信息
    // 以及详细的用卡信息以文本文件的形式保存
    execute.findAndSaveDepositTopStudent();
}
```

构建状态方法 establishState 分别调用了三个子方法，分别实现读取学生数据，读取一卡通数据并建立映射，读取消费数据并保存。具体实现如下：

```
public void establishState() {
    try {
        this.loadStudentData();
        this.loadCardData();
        this.loadRecordData();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
// 读取学生数据
```

```
private void loadStudentData() throws IOException {
    FileManager fileManager = new FileManager(STUDENT_DATA_FILE,
                                              FileManager.IOMode.READ);
    fileManager.readLines(line -> {
        String[] fields = line.trim().split(" ");
        String studentId = fields[0], name = fields[1];
        String major = fields[3], phone = fields[4];
        int year = Integer.parseInt(fields[2]); // 入学年份
        container.addStudent(studentId, name, year, major, phone);
    });
}
// 读取一卡通数据并建立映射
private void loadCardData() throws IOException {
    FileManager fileManager = new FileManager(CARD_DATA_FILE,
                                            FileManager.IOMode.READ);
    fileManager.readLines(line -> {
        String[] fields = line.trim().split(" ");
        // 学号 一卡通卡号
        String studentId = fields[0], cardId = fields[1];
        container.linkStudentCard(studentId, cardId);
    });
}
// 读取消费数据并保存
private void loadRecordData() throws IOException {
    FileManager fileManager = new FileManager(RECORD_DATA_FILE,
FileManager.IOMode.READ);
    fileManager.readLines(line -> {
        String[] fields = line.trim().split(" ");
        String cardId = fields[0];  // 卡号
        // 用卡类型
        Record.RecordType type = (Integer.parseInt(fields[1])==0) ?
                                  Record.RecordType.DEPOSIT :
                                  Record.RecordType.CONSUME;
        double amount = Double.parseDouble(fields[2]);  // 发生金额
        int place = Integer.parseInt(fields[3]);    // 发生地点编号
        container.addRecord(cardId, type, amount, place);
    });
}
```

统计每位学生一卡通的消费情况方法中，首先使用 sort 方法按消费金额的非降序对学生进行排序，之后使用 forEach 方法将排序后的学生基本信息、其对应的一卡通基本信息和消费总金额发送到屏幕显示：

```
public void statisticsAndSortStudent() {
    // 按消费金额的非降序对学生进行排序
    this.container.getStudents().sort((student1, student2) -> {
```

```
            int consume1=(int)(container.findCardByStudentId(student1.getId()).getConsume()*100);
            int consume2=(int)(container.findCardByStudentId(student2.getId()).getConsume()*100);
            return consume1 - consume2;
        });
        //将排序后的学生基本信息、其对应的一卡通基本信息和消费总金额发送到屏幕显示
        container.getStudents().forEach(student -> {
            System.out.println(SPLIT_STRING);
            System.out.println(student);
            System.out.println(container
                               .findCardByStudentId(student.getId())
                               .briefInformation());
        });
    }
```

第三步中采用遍历的方法找到存钱最多的学生，并将其基本信息、其对应的一卡通信息，以及详细的用卡信息以文本文件的形式保存，具体实现如下：

```
public void findAndSaveDepositTopStudent() {
    // 找到存钱最多的学生
    List<Student> students = this.container.getStudents();
    Student depositTopStudent = students.get(0);
    for (Student student : students) {
        if (this.container.findCardByStudentId(
            depositTopStudent.getId()).getDeposit()<this.container
            .findCardByStudentId(student.getId()).getDeposit()) {
           depositTopStudent = student; // 若当前学生的存钱更多，替换之
        }
    }
    //将其基本信息、其对应的一卡通信息，以及详细的用卡信息以文本文件的形式保存
    FileManager fileManager = new FileManager(SAVE_TO_FILE,
                                              FileManager.IOMode.WRITE);
    try {
        fileManager.writeLines(SPLIT_STRING,
          depositTopStudent.toString(),
          container.findCardByStudentId(depositTopStudent.getId()).detailedInformation());
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

四、实验结果

运行 Execute 类，得到控制台输出：

```
**************************************************
学号：20188374858604
姓名：宋和科
入学年份：2018
```

专业：电器工程
联系方式：133279242
卡号：0000000008
余额：23.06
该卡共消费：0.00
****
学号：20181028945381
姓名：黄费句
入学年份：2018
专业：计算机科学
联系方式：158621799
卡号：0000000004
余额：32.00
该卡共消费：69.15
****
学号：20171243670365
姓名：黄法框
入学年份：2017
专业：材料化学
联系方式：130156195
卡号：0000000002
余额：182.61
该卡共消费：190.30
****
学号：20188374853727
姓名：黄律气
入学年份：2018
专业：电器工程
联系方式：130156547
卡号：0000000006
余额：371.60
该卡共消费：347.40
****
学号：20188374859640
姓名：宋法和
入学年份：2018
专业：电器工程
联系方式：158621593
卡号：0000000010
余额：803.04
该卡共消费：377.24
****
学号：20188374858693
姓名：张框

入学年份：2018
专业：电器工程
联系方式：130156798
卡号：0000000009
余额：350.12
该卡共消费：391.88
****
学号：20161028949800
姓名：李法
入学年份：2016
专业：计算机科学
联系方式：130156728
卡号：0000000001
余额：392.51
该卡共消费：465.36
****
学号：20181028475485
姓名：王各件
入学年份：2018
专业：信息管理
联系方式：138126140
卡号：0000000003
余额：1011.35
该卡共消费：569.68
****
学号：20181028947850
姓名：孙大
入学年份：2018
专业：计算机科学
联系方式：130156255
卡号：0000000005
余额：822.62
该卡共消费：682.22
****
学号：20188374855641
姓名：李国
入学年份：2018
专业：电器工程
联系方式：130156761
卡号：0000000007
余额：207.21
该卡共消费：976.55
文件 SaveTop.txt 内容：
****

学号：20181028475485
姓名：王各件
入学年份：2018
专业：信息管理
联系方式：138126140
卡号：0000000003
余额：1011.35
存钱 金额：11.51 消费地点：1
存钱 金额：36.23 消费地点：2
存钱 金额：36.74 消费地点：1
存钱 金额：23.21 消费地点：0
消费 金额：12.89 消费地点：1
存钱 金额：40.22 消费地点：0
消费 金额：9.27 消费地点：2
存钱 金额：47.19 消费地点：1
消费 金额：25.54 消费地点：0
消费 金额：10.16 消费地点：3
存钱 金额：41.87 消费地点：1
存钱 金额：13.10 消费地点：3
存钱 金额：22.35 消费地点：3
消费 金额：33.65 消费地点：1
存钱 金额：47.70 消费地点：3
存钱 金额：3.38 消费地点：2
存钱 金额：8.59 消费地点：3
消费 金额：12.11 消费地点：1
存钱 金额：36.59 消费地点：0
存钱 金额：26.32 消费地点：3
存钱 金额：26.61 消费地点：0
存钱 金额：27.54 消费地点：0
消费 金额：38.05 消费地点：0
消费 金额：6.60 消费地点：3
存钱 金额：15.98 消费地点：0
存钱 金额：47.77 消费地点：0
存钱 金额：41.17 消费地点：0
消费 金额：48.74 消费地点：0
存钱 金额：11.27 消费地点：2
存钱 金额：44.93 消费地点：0
存钱 金额：44.83 消费地点：1
存钱 金额：2.76 消费地点：2
消费 金额：6.92 消费地点：2
存钱 金额：46.58 消费地点：2
消费 金额：40.36 消费地点：1
存钱 金额：32.72 消费地点：1
存钱 金额：23.54 消费地点：2

存钱 金额：4.51 消费地点：3
存钱 金额：17.78 消费地点：2
存钱 金额：38.52 消费地点：0
存钱 金额：44.44 消费地点：1
消费 金额：24.20 消费地点：0
存钱 金额：42.70 消费地点：2
存钱 金额：10.13 消费地点：2
消费 金额：28.91 消费地点：3
存钱 金额：16.13 消费地点：1
存钱 金额：32.25 消费地点：1
存钱 金额：42.94 消费地点：1
消费 金额：26.71 消费地点：3
存钱 金额：3.01 消费地点：1
存钱 金额：26.56 消费地点：1
消费 金额：16.52 消费地点：1
消费 金额：37.37 消费地点：1
存钱 金额：46.62 消费地点：3
存钱 金额：9.60 消费地点：1
存钱 金额：45.89 消费地点：1
消费 金额：34.15 消费地点：2
消费 金额：7.16 消费地点：3
存钱 金额：22.97 消费地点：2
存钱 金额：6.79 消费地点：1
消费 金额：7.67 消费地点：0
消费 金额：17.52 消费地点：2
存钱 金额：34.90 消费地点：2
存钱 金额：1.85 消费地点：0
存钱 金额：24.36 消费地点：3
消费 金额：17.79 消费地点：1
消费 金额：21.83 消费地点：3
存钱 金额：14.85 消费地点：3
存钱 金额：0.04 消费地点：0
消费 金额：30.27 消费地点：3
消费 金额：22.96 消费地点：0
存钱 金额：20.71 消费地点：0
存钱 金额：27.28 消费地点：0
存钱 金额：31.86 消费地点：3
消费 金额：18.40 消费地点：3
存钱 金额：11.76 消费地点：1
消费 金额：13.93 消费地点：1
存钱 金额：46.62 消费地点：1
存钱 金额：21.03 消费地点：3
存钱 金额：43.60 消费地点：0
存钱 金额：43.69 消费地点：0

存钱 金额：24.09 消费地点：0
存钱 金额：28.90 消费地点：2
存钱 金额：33.95 消费地点：1
该卡共消费：569.68

五、实验总结

通过本次实验，我掌握了掌握类的基础知识，熟悉使用了文件 IO 的使用；对集合类，以及 Java 的基本数据结构的使用有了一定的认识。