西南民族大学

**实验报告**

20 20 ------20 21 学年第1学期

课程名称：软件工程课程设计

学院：计科学院 专业：软件工程

年级： 2018 班级：1801

学号： 201831104068 姓名：徐士吉

同组人：唐昭洋

|  |
| --- |
| 西南民族大学学生实验报告  教学单位：计科学院 实验室名称：bs-223 实验时间：2020年9月28日  姓名：徐士吉 专业： 软件工程 班级：1801 学号:201831104068 |
| 实验项目名称： 文件处理 实验成绩： 教师签名：周旭川 |
| 实验项目报告内容（1、实验背景（目的、意义及原理等）；2、材料与方法；3、实验主要过程与结果；  4、分析讨论；5、教师评阅）。   1. 实验背景   实验目的：  （1）单独运行yq.exe或者yq.java，给出缺省输入文件的输出； （2）运行yq [指定输入文件] [指定输出文件]；（按要求排序） （3）运行yq [指定输入文件][指定输出文件] [指定省份]；（按要求排序） 把个人项目的三个的功能封装为独立模块，并设计单元测试。   1. 材料与方法： git，java编译器 2. 主要过程与方法 3. 解决思路： ①在原有实验的基础上将函数初步进行封装，参数为2（[指定输入文件] [指定输出文件]）的一个函数与参数为3（[指定输入文件][指定输出文件] [指定省份]）的一个函数。 ②将读入输入文件的内容写入list并将其归入某一函数，后续可调用。 ③将按照人数总数对省份进行排序的这一段单独封装成一个函数。 ④将各个省份内的城市排序的这一段单独封装成一个函数。 4. 解决代码：   java：  **package** 文件处理封装;  **import** java.io.File;  **import** java.io.FileReader;  **import** java.io.FileWriter;  **import** java.util.ArrayList;  **import** java.util.LinkedList;  **import** java.util.List;  **import** java.util.Scanner;  **public** **class** wenjiianmain {  **public** **static** **void** main(String[] args) {  Scanner in = **new** Scanner(System.***in***);  String input = in.nextLine();  Operation operation = **new** Operation();  operation.*match*(input).work();  in.close();  }  }  **abstract** **class** Basic {  **public** **void** work() {  };  }  **class** Operation {  **public** **static** Basic match(String input) {  String[] splitinput = input.split(" ");  **switch** (splitinput.length) {  **case** 1:  **return** **new** File\_work();  **case** 2:  **return** **new** File\_sort(splitinput[0], splitinput[1]);  **case** 3:  **return** **new** File\_city\_sort(splitinput[0], splitinput[1], splitinput[2]);  **default**:  System.***out***.println("输入错误");  **return** **null**;  }  }  }  **class** File\_work **extends** Basic {  **public** **void** work() {  File file = **new** File("E://yq\_in.txt");  FileReader fr = **null**;  FileWriter fw = **null**;  **try** {  fr = **new** FileReader(file);  **char**[] data = **new** **char**[1500];  **int** length = 0;  String str = **null**;  **while** ((length = fr.read(data)) > 0) {  str = **new** String(data, 0, length);  }  String[] splitLine = str.split("\n");  List<String> newLine = **new** LinkedList<>();  **for** (**int** i = 0; i < splitLine.length - 1; i++) {  String province = splitLine[i].substring(0, 3);  String nextProvince = splitLine[i + 1].substring(0, 3);  **if** (i == 0)  newLine.add(province);  newLine.add(splitLine[i].substring(splitLine[i].indexOf("\t") + 1, splitLine[i].length()));  **if** (province.equals(nextProvince) == **false**) {  newLine.add("\n");  newLine.add(nextProvince);  }  }  StringBuilder sb = **new** StringBuilder();  **for** (**int** i = 0; i < newLine.size(); i++) {  sb.append(newLine.get(i));  **if** (newLine.get(i).equals("\n") == **false**) {  sb.append("\n");  }  }  String s = sb.toString();  fw = **new** FileWriter("E://yq\_out.txt");  fw.write(s);  } **catch** (Exception e) {  e.printStackTrace();  } **finally** {  **try** {  fr.close();  fw.close();  } **catch** (Exception e) {  e.printStackTrace();  }  }  }  }  **class** File\_sort **extends** Basic {  **private** String txt\_in, txt\_out;  File\_sort(String txt\_in, String txt\_out) {  **this**.txt\_in = txt\_in;  **this**.txt\_out = txt\_out;  }  **public** **void** work() {  File file = **new** File("E://" + txt\_in + ".txt");  FileReader fr = **null**;  FileWriter fw = **null**;  **try** {  fr = **new** FileReader(file);  **char**[] data = **new** **char**[1500];  **int** length = 0;  String str = **null**;  **while** ((length = fr.read(data)) > 0) {  str = **new** String(data, 0, length);  }  String[] splitstr = str.split("\r\n|\t");  ExchangeClass[] exchangeclass = **new** ExchangeClass[20];  **for** (**int** i = 0; i < exchangeclass.length; i++) {  exchangeclass[i] = **new** ExchangeClass();  }  **int** mark\_tmp = 0;  **int** index = 0;  **int** city\_tmp = 0;  **boolean** bool\_index = **true**;  **for** (**int** i = 0; i < splitstr.length / 3 - 1; i++) {  String province = splitstr[3 \* i];  String nextProvince = splitstr[3 \* (i + 1)];  exchangeclass[index].city.add(splitstr[3 \* i + 1]);  exchangeclass[index].mark.add(splitstr[3 \* i + 2]);  city\_tmp++;  mark\_tmp += Integer.*parseInt*(splitstr[3 \* i + 2]);  **if** (province.equals(nextProvince) == **false**) {  exchangeclass[index].province = splitstr[3 \* i];  exchangeclass[index].total\_city = city\_tmp;  exchangeclass[index].total\_mark = mark\_tmp;  index++;  mark\_tmp = 0;  city\_tmp = 0;  }  **if** (i == splitstr.length / 3 - 2) {  **if** (province.equals(nextProvince) == **false**) {  bool\_index = **true**;  exchangeclass[index].province = splitstr[3 \* (i + 1)];  exchangeclass[index].total\_city = 1;  exchangeclass[index].total\_mark = Integer.*parseInt*(splitstr[3 \* (i + 1) + 2]);  exchangeclass[index].city.add(splitstr[3 \* (i + 1) + 1]);  exchangeclass[index].mark.add(splitstr[3 \* (i + 1) + 2]);  } **else** {  bool\_index = **false**;  exchangeclass[index - 1].total\_city += 1;  exchangeclass[index - 1].total\_mark += Integer.*parseInt*(splitstr[3 \* (i + 1) + 2]);  exchangeclass[index - 1].city.add(splitstr[3 \* (i + 1) + 1]);  exchangeclass[index - 1].mark.add(splitstr[3 \* (i + 1) + 2]);  }  }  }  **if** (bool\_index == **false**)  index--;  **for** (**int** i = 0; i <= index; i++) {  **for** (**int** j = 0; j <= index - i - 1; j++) {  **if** (exchangeclass[j].total\_mark < exchangeclass[j + 1].total\_mark) {  ExchangeClass tmp = **new** ExchangeClass();  tmp = exchangeclass[j];  exchangeclass[j] = exchangeclass[j + 1];  exchangeclass[j + 1] = tmp;  }  **if** (exchangeclass[j].total\_mark == exchangeclass[j + 1].total\_mark) {  **if** (exchangeclass[j].province.compareTo(exchangeclass[j + 1].province) > 0) {  ExchangeClass tmp = **new** ExchangeClass();  tmp = exchangeclass[j];  exchangeclass[j] = exchangeclass[j + 1];  exchangeclass[j + 1] = tmp;  }  }  }  }  **for** (**int** i = 0; i <= index; i++) {  **for** (**int** j = 0; j < exchangeclass[i].total\_city; j++) {  **for** (**int** k = 0; k < exchangeclass[i].total\_city - j - 1; k++) {  **if** (Integer.*parseInt*(exchangeclass[i].mark.get(k)) < Integer  .*parseInt*(exchangeclass[i].mark.get(k + 1))) {  String tmpcity, tmpmark;  tmpcity = exchangeclass[i].city.get(k);  exchangeclass[i].city.set(k, exchangeclass[i].city.get(k + 1));  exchangeclass[i].city.set(k + 1, tmpcity);  tmpmark = exchangeclass[i].mark.get(k);  exchangeclass[i].mark.set(k, exchangeclass[i].mark.get(k + 1));  exchangeclass[i].mark.set(k + 1, tmpmark);  }  **if** (Integer.*parseInt*(exchangeclass[i].mark.get(k)) == Integer  .*parseInt*(exchangeclass[i].mark.get(k + 1))) {  **if** (exchangeclass[i].city.get(k).compareTo(exchangeclass[i].city.get(k + 1)) > 0) {  String tmpcity, tmpmark;  tmpcity = exchangeclass[i].city.get(k);  exchangeclass[i].city.set(k, exchangeclass[i].city.get(k + 1));  exchangeclass[i].city.set(k + 1, tmpcity);  tmpmark = exchangeclass[i].mark.get(k);  exchangeclass[i].mark.set(k, exchangeclass[i].mark.get(k + 1));  exchangeclass[i].mark.set(k + 1, tmpmark);  }  }  }  }  }  StringBuilder sb = **new** StringBuilder();  **for** (**int** i = 0; i <= index; i++) {  sb.append(exchangeclass[i].province + " " + exchangeclass[i].total\_mark + "\n");  **for** (**int** j = 0; j < exchangeclass[i].total\_city; j++)  sb.append(exchangeclass[i].city.get(j) + " " + exchangeclass[i].mark.get(j) + "\n");  sb.append("\n");  }  String s = sb.toString();  fw = **new** FileWriter("E://" + txt\_out + ".txt");  fw.write(s);  } **catch** (Exception e) {  e.printStackTrace();  } **finally** {  **try** {  fr.close();  fw.close();  } **catch** (Exception e) {  e.printStackTrace();  }  }  }  }  **class** File\_city\_sort **extends** Basic {  **private** String txt\_in, txt\_out, sortcity;  File\_city\_sort(String txt\_in, String txt\_out, String sortcity) {  **this**.sortcity = sortcity;  **this**.txt\_in = txt\_in;  **this**.txt\_out = txt\_out;  }  **public** **void** work() {  File file = **new** File("E://" + txt\_in + ".txt");  FileReader fr = **null**;  FileWriter fw = **null**;  **try** {  fr = **new** FileReader(file);  **char**[] data = **new** **char**[1500];  **int** length = 0;  String str = **null**;  **while** ((length = fr.read(data)) > 0) {  str = **new** String(data, 0, length);  }  String[] splitstr = str.split("\r\n|\t");  **int** marktmp = 0;  **int** totalcitytmp = 0;  ExchangeClass exchangeclass = **new** ExchangeClass();  **for** (**int** i = 0; i < splitstr.length / 3; i++) {  **if** (sortcity.equals(splitstr[3 \* i])) {  exchangeclass.city.add(splitstr[3 \* i + 1]);  exchangeclass.mark.add(splitstr[3 \* i + 2]);  totalcitytmp++;  marktmp += Integer.*parseInt*(splitstr[3 \* i + 2]);  }  }  exchangeclass.total\_city = totalcitytmp;  exchangeclass.province = sortcity;  exchangeclass.total\_mark = marktmp;  **for** (**int** i = 0; i < exchangeclass.total\_city; i++) {  **for** (**int** j = 0; j < exchangeclass.total\_city - i - 1; j++) {  **if** (Integer.*parseInt*(exchangeclass.mark.get(j)) < Integer.*parseInt*(exchangeclass.mark.get(j + 1))) {  String tmpcity, tmpmark;  tmpcity = exchangeclass.city.get(j);  exchangeclass.city.set(j, exchangeclass.city.get(j + 1));  exchangeclass.city.set(j + 1, tmpcity);  tmpmark = exchangeclass.mark.get(j);  exchangeclass.mark.set(j, exchangeclass.mark.get(j + 1));  exchangeclass.mark.set(j + 1, tmpmark);  }  **if** (Integer.*parseInt*(exchangeclass.mark.get(j)) == Integer  .*parseInt*(exchangeclass.mark.get(j + 1))) {  **if** (exchangeclass.city.get(j).compareTo(exchangeclass.city.get(j + 1)) > 0) {  String tmpcity, tmpmark;  tmpcity = exchangeclass.city.get(j);  exchangeclass.city.set(j, exchangeclass.city.get(j + 1));  exchangeclass.city.set(j + 1, tmpcity);  tmpmark = exchangeclass.mark.get(j);  exchangeclass.mark.set(j, exchangeclass.mark.get(j + 1));  exchangeclass.mark.set(j + 1, tmpmark);  }  }  }  }  StringBuilder sb = **new** StringBuilder();  sb.append(exchangeclass.province + " " + exchangeclass.total\_mark + "\n");  **for** (**int** i = 0; i < exchangeclass.total\_city; i++) {  sb.append(exchangeclass.city.get(i) + " " + exchangeclass.mark.get(i) + "\n");  }  String s = sb.toString();  fw = **new** FileWriter("E://" + txt\_out + ".txt");  fw.write(s);  } **catch** (Exception e) {  e.printStackTrace();  } **finally** {  **try** {  fr.close();  fw.close();  } **catch** (Exception e) {  e.printStackTrace();  }  }  }  }  **class** ExchangeClass {  **public** String province;  **public** **int** total\_mark;  **public** **int** total\_city;  **public** ArrayList<String> city = **new** ArrayList<>();  **public** ArrayList<String> mark = **new** ArrayList<>();  }   1. 实验结果截图：   ①不输入默认输出：      ②不指定省排序输出：      ③指定省份排序输出：       1. 分析讨论   通过此次实验，我进一步掌握了如何用变成语言处理数据和对该过程封装处理，同时通过此次实验，我进一步掌握了处理数据语言的方法。 |

注：实验报告的内容及格式可由学院根据学科专业特点确定；全校各专业必须使用学校统一封面。