西南民族大学

**实验报告**

2019 ------2020 学年第 2 学期

课程名称：软件工程课程设计

学 院：计算机科学与技术

专 业：计算机科学与技术

年 级：2017级

班 级：1702班

姓 名：麦蔼童

学 号：201731007023

|  |
| --- |
| 西南民族大学学生实验报告  教学单位：计算机科学与技术 实验室名称：BS-226 实验时间： 2020年3月18日  专业：计算机科学与技术 姓名：麦蔼童 学号：201731007023 班级：1702班 |
| 实验项目名称：第四次实验 实验成绩： 教师签名：周绪川 |
| 一、实验要求  在实验三的基础上继续：  输入文件为yq\_in\_04.txt，输出文件yq\_out\_04.txt,但有如下要求：  1、每个省后面有一个总数;  2、输出省按总数从大到小排序；如果两个省总数一样，按拼音（字母）排序；  3、每个省内各市从大到小排序；如果两个市总数一样，按拼音（字母）排序；  二、实验内容  1、解决思路  文本处理方面：  ①用二维数组data[ ][ ]存储数据。  将输入数据逐行处理，将每一行主观上视为两个组成部分：“省份”+“地区和数量”。考虑到数据的数量不算庞大，用二维数组data[ ][ ]存储数据，每个省份的数据用一行来存储，其中data[][0]用于存储每个省份的分支数目（即地区数目），便于输出；而data[][1]用于存储省份信息，便于处理省份；从data[][2]开始存储地区数目。逐行处理，每行处理时，先对比省份信息，相同即存储地区信息；不相同即数组在新的一行再作处理。注意：第一个省份处理时，需要特殊处理。  ②使用java.text.Collator排序。  该类使用字符串包含字符在指定语言的自然顺序（譬如中文汉字的自然顺序）做比较。每处理完一个省份，就将其下的地区进行排序。由于需要按照中文的拼音来排序，java内置的排序引擎并不能实现，因此，这里用到Collator写一个工具类方法对中文进行排序。  ③数字的处理。  由于需要根据每个地区的数量进行排序，新建一个int型数组num[i][j]存储对应data[i][j]地区的数量以及省份总数。int型数字比较大小方便一点，同时方便统计省份data[i][1]的总数存储在对应的num[i][1]。  ④省份顺序处理  由于是利用二维数组每一行代表一个省份，如果在排序的过程中，同步更新data数组的顺序的话，需要大块数据一起交换，因此，新建一个order数组，用来记录省份的顺序，在输出的时候按照order记录的顺序，输出data数组的内容即可。  实现命令行指定输入输出文件：  首先，命令行输入的参数存储在main参数args数组中。因此输入的三个参数分别存储在args[0]（输入文件）、args[1]（输出文件）、args[2]（指定省份）中。然后利用File类来创建指定文件名的实例，再通过FileReader、FileWriter来进行文件的读取和存储。  先用ecilpse完成，导出项目后，生成jar文件后，再用exej4转化成可执行文件yq.exe  2、流程    **图1 流程图**  3、代码及结果测试  **import** java.io.\*  **public** **class** yq\_4 {  **public** **static** String[][] *data* = **new** String[100][100];  **public** **static** **int**[][] *num* = **new** **int** [100][100]; // 记录省份及地区的数量 （以整形形式，便于比较）    **public** **static** **void** sortByChinese(List<String> values) {  **final** Collator pyComparator = Collator.*getInstance*(Locale.***CHINA***); // 中文排序关键对象  Collections.*sort*(values,**new** Comparator<String>() {    @Override  **public** **int** compare(String s1,String s2) {  **int** result = pyComparator.compare(s1, s2);  **return** result;  }  });  }  **public** **static** **void** sort(**int** i,**int** j) { // i是哪个省份 j是省份的地区数  **int** k,p;  **for** (k=3; k<j; k++) { // 处理上一省份的地区排序  **for** (p=2; p<j-i; p++) {  **if** (*num*[i][p]<*num*[i][p+1]) {  **int** t1 = *num*[i][p]; // 交换数字  *num*[i][p] = *num*[i][p+1];  *num*[i][p+1] = t1;  String t2 = *data*[i][p]; // 交换地区名  *data*[i][p] = *data*[i][p+1];  *data*[i][p+1] = t2;  }  **if** (*num*[i][p]==*num*[i][p+1]) { // 地区数字相等 则按地区名的拼音排序  List<String> list = **new** ArrayList<String>();  list.add(*data*[i][p]);  list.add(*data*[i][p+1]);  *sortByChinese*(list);  String[] str = list.toArray(**new** String[list.size()]);  **if** (str[0].equals(*data*[i][p])==**false**) {  **int** t1 = *num*[i][p]; // 交换数字  *num*[i][p] = *num*[i][j];  *num*[i][p+1] = t1;  String t2 = *data*[i][p]; // 交换地区名  *data*[i][p] = *data*[i][p+1];  *data*[i][p+1] = t2;  }  }  }  }  }    **public** **static** **void** main(String[] args) {    // 参数存储在args数组中  System.***out***.println(args.length);  **for** (String s : args) {  System.***out***.println(s + "\t");  }  BufferedReader br = **null**;  BufferedWriter bw = **null**;  // String[][] data = new String[100][100];  // int[][] num = new int [100][100]; // 记录省份及地区的数量 （以整形形式，便于比较）  **int**[] order = **new** **int**[100]; // 记录省份输出顺序  **int** tag = 0;  **int** i, j, k, p;    **try** {  File path = **new** File("D:/yqtest"); // 读入文件的路径  File in = **new** File(path, args[0]);  File out = **new** File(path, args[1]);  i = 0;  j = 2;  *data*[0][1] = "";  FileReader fr = **new** FileReader(in);  br = **new** BufferedReader(fr);    String s = "";// 储存输入文件中的每行  **while** ((s = br.readLine()) != **null**) {  s = s.trim();  **int** pos = s.indexOf('省'); // 定位省份  **if** (*data*[i][1].equals(s.substring(0, pos + 1))) { // 与上一处理行同一个省份  *data*[i][j] = s.substring(pos + 2); // 保存地区  **int** posn = *data*[i][j].indexOf('\t');  *num*[i][j] = Integer.*parseInt*(*data*[i][j].substring(posn+1)); // 保存地区对应数字  *num*[i][1] += *num*[i][j]; // 统计各个省份总数  j = j + 1;  }  **else** { // 与上一处理行不同省份  **if** (tag==0) { // tag=0 是第一个省份  *data*[i][1] = s.substring(0, pos + 1);  *data*[i][j] = s.substring(pos + 2);  **int** posn = *data*[i][j].indexOf('\t');  *num*[i][j] = Integer.*parseInt*(*data*[i][j].substring(posn+1)); // 保存地区对应数字  *num*[i][1] += *num*[i][j]; // 统计各个省份总数  j = j + 1;  tag = 1;  }  **else** {// 不是第一个省份  *data*[i][0] = j + ""; // 记录上一个处理完成省份的分支个数  *data*[i][1] += "\t" + String.*valueOf*(*num*[i][1]); // 在省份后面连接总数 方便输出  *sort*(i,j-1);  i = i + 1;  j = 2;  *data*[i][1] = s.substring(0, pos + 1);  *data*[i][j] = s.substring(pos + 2);  **int** posn = *data*[i][j].indexOf('\t');  *num*[i][j] = Integer.*parseInt*(*data*[i][j].substring(posn+1)); // 保存地区对应数字  *num*[i][1] += *num*[i][j]; // 统计各个省份总数  j = j + 1;  }  }  }  *data*[i][0] = j + "";  *data*[i][1] += "\t" + String.*valueOf*(*num*[i][1]);  *sort*(i,j); //处理最后一个省的排序  **int** row = i+1; // 省份总个数  **int** num1[] = **new** **int**[100];  **for** (k=0;k<row;k++) { //初始化order数组  order[k] = k;  num1[k] = *num*[k][1];  }    **for** (k=1; k<row; k++) { // 处理省份的排列  **for** (p=0; p<row-k; p++) {  **if** (*num*[order[p]][1]<*num*[order[p+1]][1]) {  **int** t = order[p];  order[p] = order[p+1];  order[p+1] = t;  }  **if** (*num*[order[p]][1]==*num*[order[p+1]][1]) {  List<String> list = **new** ArrayList<String>();  list.add(*data*[order[p]][1]);  list.add(*data*[order[p+1]][1]);  *sortByChinese*(list);  String[] str = list.toArray(**new** String[list.size()]);  **if** (str[0].equals(*data*[order[p]][1])==**false**) {  **int** t = order[p];  order[p] = order[p+1];  order[p+1] = t;  }  }  }  }    FileWriter fw = **new** FileWriter(out);  bw = **new** BufferedWriter(fw);  **if** (args.length == 3) { //指定省份时，无关顺序  **for** (i = 0; i < row; i++) {  **if** (args[2].equals(*data*[i][1])) {  bw.write(*data*[i][1]);  bw.newLine();  **for** (j = 2; j < Integer.*parseInt*(*data*[i][0]); j++) {  bw.write(*data*[i][j]);  bw.newLine();  }  **break**;  }  }  } **else** { // 全部输出时 要按从大到小顺序输出  **for** (i = 0; i < row; i++) {  bw.write(*data*[order[i]][1]);  bw.newLine();  **for** (j = 2; j < Integer.*parseInt*(*data*[order[i]][0]); j++) {  bw.write(*data*[order[i]][j]);  bw.newLine();  }  bw.newLine();  }  }  bw.flush();  **for** (i=0;i<row;i++) { // 检验省份排序  System.***out***.print(order[i] + "\t");  }  } **catch** (Exception e) {  e.printStackTrace();  } **finally** {  **try** {  bw.close();  br.close();  } **catch** (Exception e) {  e.printStackTrace();  }  }  System.***out***.println("Completed!");  }  }  测试结果：  生成可执行文件：    **图2**  **不指定省份时：**    **图3**    **图4**  **指定省份时：**    **图5**    **图6**  4、总结  本次实验在实验三基础上增加了排序，而是涉及到中文字符的排序，由于对Java并不算很熟练，通过查资料发现很多Java内置的比较引擎和排序引擎都不能满足按照中文拼音排序。比如说String.compare等的字符串比较以及collections.sort等list的排序。因此，用到Collator写一个工具类方法对中文进行排序，创建一个中文排序的关键对象：**final** Collator pyComparator = Collator.*getInstance*(Locale.***CHINA***);但是这个方法是将List排序，返回的也是一个List，而我本身储存信息的是data数组，因此需要进行Array转List，排序后再List转回Array，由于只是两个中文字符进行排序，因此这样转化也不会太麻烦，但还可以思考如何优化。  通过这几次实验，提升了自己的思考问题和寻求解决方法的能力，锻炼了自己的思维能力，并且对Java更加熟悉了。 |

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。