西南民族大学

**实验报告**

**2021------2022**学年第**2**学期

课程名称：软件工程课程设计

学院：计算机科学与工程

专业：计算机科学与技术

年级：2019级 班级：1902

学号：201931101200 姓名：索比亚

同组人：索比亚 201931101010

|  |
| --- |
| 西南民族大学学生实验报告  教学单位：计算机科学与工程学院 实验室名称：BS-223 实验时间：2022 年 3 月 27 日  姓名：索比亚，卜亚楠 专业：计算机科学与技术 班级：1902 学号:201931101200、201931101010 |
| 实验项目名称：实验3，4 实验成绩： 教师签名：周绪川 |
| 1. **实验目的**   **结对编程:**  **我们已经实现了三个功能:。**  **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**  **要求：**  **语言要求：C/C++，Java，。。。**  **实验报告内容：解决思路，流程，代码及结果测试；**  **源码上传至个人git.**  **二、材料与方法**  PC机一台、java语言编程环境，exe4j打包软件   1. **实验主要过程与结果** 2. **配置实验环境**   （1）语言要求：开发语言Java  （2）IDE：idea     1. **编写代码**   思路：将三个功能封装成类中的方法，便于可以独立调用  第一步：将文件输入输出流的创建方法封装成静态方法，方便接下来进行调用  第二步：创建操作文件类，创建私有属性，输入输出流，创建构造函数要求在创建类的时候必须传入输入输出流参数。  第三步：创建解析文件的方法，利用我的类中的输入流获取数据，并把数据封装为LinkedHashMap返回.  第三步：创建解析文件的方法，利用我的类中的输入流获取数据，并把数据封装为  第四步：创建私有方法对Map数据进行排序，只允许本类调用，维护数据安全。  第五步：创建对数据的排序处理方法，外界传入LinkedHashMap<String, LinkedHashMap<String,Integer>>类型的数据（即解析文件获取的数据类型）即可对数据进行排序，严格规定数据的泛型，防止传入的数据无法处理  第六步：创建根据省份返回数据的方法，传入LinkedHashMap<String, LinkedHashMap<String,Integer>>类型的数据，以及String类型的省份名称获取数据，返回值依然为LinkedHashMap<String, LinkedHashMap<String,Integer>>类型的数据  第七步：以文件输出流的方式输出信息，传入LinkedHashMap<String, LinkedHashMap<String,Integer>>类型的数据，将信息输出到输出流的地址中，将数据处理与输出隔离开，方便扩展与维护  第八步：虽然java具有垃圾回收机制，但是不要忘记关闭我们的输入输出流。  第九步：利用maven工具将文件打包问jar文件，方便其他项目的调用               1. **运行结果**   第一步：单元测试  Maven导入单元测试包  <**dependencies**>  <**dependency**>  <**groupId**>junit</**groupId**>  <**artifactId**>junit</**artifactId**>  <**version**>4.12</**version**>  </**dependency**>  </**dependencies**>  创建单元测试代码  **import** java.io.\*;  **import** java.util.LinkedHashMap;  **class** GetFileTest {  @org.junit.jupiter.api.Test  **void** getFileData() {  BufferedReader bufferReader = **null**;  OutputStreamWriter bufferWriter = **null**;  **try** {  bufferReader = FileUtils.*getBufferReader*(**"F:\\yq\_in.txt"**);  bufferWriter = FileUtils.*getBufferWriter*(**"F:\\yq\_out.txt"**);  } **catch** (FileNotFoundException e) {  e.printStackTrace();  } **catch** (UnsupportedEncodingException e) {  e.printStackTrace();  }  OperationFile getFile = **new** OperationFile(bufferReader,bufferWriter);  LinkedHashMap<String, LinkedHashMap<String, Integer>> fileData = **null**;  **try** {  fileData = getFile.getFileData();  } **catch** (IOException e) {  e.printStackTrace();  }  System.***out***.println(fileData);  }  @org.junit.jupiter.api.Test  **void** sortData() {  BufferedReader bufferReader = **null**;  OutputStreamWriter bufferWriter = **null**;  **try** {  bufferReader = FileUtils.*getBufferReader*(**"F:\\yq\_in.txt"**);  bufferWriter = FileUtils.*getBufferWriter*(**"F:\\yq\_out.txt"**);  } **catch** (FileNotFoundException e) {  e.printStackTrace();  } **catch** (UnsupportedEncodingException e) {  e.printStackTrace();  }  OperationFile getFile = **new** OperationFile(bufferReader,bufferWriter);  LinkedHashMap<String, LinkedHashMap<String, Integer>> fileData = **null**;  **try** {  fileData = getFile.getFileData();  } **catch** (IOException e) {  e.printStackTrace();  }  LinkedHashMap<String, LinkedHashMap<String, Integer>> sortData = getFile.sortData(fileData);  System.***out***.println(sortData);  }  @org.junit.jupiter.api.Test  **void** getByProvince() {  BufferedReader bufferReader = **null**;  OutputStreamWriter bufferWriter = **null**;  **try** {  bufferReader = FileUtils.*getBufferReader*(**"F:\\yq\_in.txt"**);  bufferWriter = FileUtils.*getBufferWriter*(**"F:\\yq\_out.txt"**);  } **catch** (FileNotFoundException e) {  e.printStackTrace();  } **catch** (UnsupportedEncodingException e) {  e.printStackTrace();  }  OperationFile getFile = **new** OperationFile(bufferReader,bufferWriter);  LinkedHashMap<String, LinkedHashMap<String, Integer>> fileData = **null**;  **try** {  fileData = getFile.getFileData();  } **catch** (IOException e) {  e.printStackTrace();  }  LinkedHashMap<String, LinkedHashMap<String, Integer>> getByProvince = getFile.getByProvince(fileData,**"浙江省"**);  System.***out***.println(getByProvince);  }  @org.junit.jupiter.api.Test  **void** outByWriter() {  BufferedReader bufferReader = **null**;  OutputStreamWriter bufferWriter = **null**;  **try** {  bufferReader = FileUtils.*getBufferReader*(**"F:\\yq\_in.txt"**);  bufferWriter = FileUtils.*getBufferWriter*(**"F:\\yq\_out.txt"**);  } **catch** (FileNotFoundException e) {  e.printStackTrace();  } **catch** (UnsupportedEncodingException e) {  e.printStackTrace();  }  OperationFile getFile = **new** OperationFile(bufferReader,bufferWriter);  LinkedHashMap<String, LinkedHashMap<String, Integer>> fileData = **null**;  **try** {  fileData = getFile.getFileData();  } **catch** (IOException e) {  e.printStackTrace();  }  **try** {  getFile.outByWriter(fileData);  } **catch** (IOException e) {  e.printStackTrace();  }  }  }  运行：    全部成功  与数据可视化结合也正常使用：      **个人git网址：**[上传文件 ·201931101200/速比屋 (github.com)](https://github.com/201931101200/suobiya/upload/main)   1. **分析讨论**   通过本次实验学到了很多比如模块化封装，他使原来零散的程序变成了可以运行的软件，模块封装化的过程比较艰难的是封装的思想，虽然出现很多问题但还要多加练习才能熟练掌握。  **五、教师评阅** |

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。