****

**面向对象程序设计上机实验报告**

实验题目3：字符串的处理

**学院名称 智能与计算学部**

**专 业 软件工程**

**学生姓名 陈昊昆**

**学 号 3021001196**

**年 级 2021级**

**班 级 3班**

**时 间 2022年 4月10日**

1. **实验目的**

熟悉Eclipse/MyEclipse；

熟悉Java中String/StringBuffer/数组的使用；

熟悉文件的读写；

1. **实验内容**

学术上目前认为红楼梦前80回作者和后40回是不同的作者，有人统计过每回中一些常见字如“之”的出现次数，认为前80回和后40回“之”字出现频率有较大差异，由此更加确定了红楼梦前后两半部分具有不同的写作风格。现在有red.txt里面存放了红楼梦120回的内容，请你通过实验来验证这种说法。程序主要完成如下功能：

1. 将读取到的红楼梦全书拆分为120回（根据第一回、第二回….第一百二十回）；
2. 统计每回中特定字符串的出现频率，记录到一个一维数组中（长度120）返回。
3. **程序实现**

**public** Analysis(String filename) **throws** Exception {

**this**.filename = filename;

}

/\*\*

\* 提示 ：将一个文本文件读取到一个字符串中返回

\*

\* **@param** filename

\* 红楼梦文本文件的全路径名

\* **@return** 文本的内容

\*/

// 以字符流的方式读入txt 并保存在String中

**private** String readFromTxt(String filename) **throws** Exception {

Reader reader = **null**;

**try** {

StringBuffer buf = **new** StringBuffer();

**char**[] chars = **new** **char**[1024];

reader = **new** InputStreamReader(**new** FileInputStream(filename), "UTF-8");

**int** readed = reader.read(chars);

**while** (readed != -1) {

buf.append(chars, 0, readed);

readed = reader.read(chars);

}

**return** buf.toString();

} **finally** {

close(reader);

}

}

/\*\*

\* 返回红楼梦中出现频率最高的N个词，频率从高到低排列（所谓词就是两个相邻的汉字）

\* **@param** n

\* **@return**

\* **@throws** Exception

\*/

**public** List<String> getTopNWords( **int** n) **throws** Exception{

List<String> list = **new** ArrayList<String>();

String f = "C:";

Analysis ana = **new** Analysis (f);

String totaltxt = "";

totaltxt = ana.readFromTxt(filename);

StringBuffer buf = **new** StringBuffer(totaltxt);

String strr = buf.toString();

**char**[] arr = strr.toCharArray();

// 剔除文中非中文的字符 防止其参与词的构成

**for**(**int** i = arr.length-1; i >= 0; i--) {

**if**(!(arr[i] >= 0x4e00)&&(arr[i] <= 0x9fbb) ||

arr[i] == '，'|| arr[i] == '．'|| arr[i] == '：'|| arr[i] == '？'

|| arr[i] == '！') {

buf.deleteCharAt(i);

}

}

totaltxt = buf.toString();

**int** txtlen = totaltxt.length();

**char** ch[] = totaltxt.toCharArray();

StringBuffer stb[]= **new** StringBuffer[txtlen-1];

String str[]= **new** String[txtlen-1];

// 构造文中所有两个字组成的词

**for**(**int** i = 0; i < txtlen-1; i++) {

stb[i] = **new** StringBuffer("");

**for**(**int** j = i; j < 2+i; j++) {

stb[i].append(ch[j]);

}

str[i] = stb[i].toString();

}

// 利用map对应词与其出现频率

Map map = **new** HashMap();

**for** (**int** i = 0; i < str.length; i++) {

**int** count = 1;

**if** (map.containsKey(str[i])) {

count = (Integer) map.get(str[i]) + 1;

}

map.put(str[i], count);

}

**for**(**int** i = 0; i < n; i++) {

**int** maxcnt = 0;

String frestr = "";

Iterator it = map.keySet().iterator();

**while**(it.hasNext()) {

String key = (String)it.next();

**if** ((Integer)map.get(key) > maxcnt) {

maxcnt = (Integer)map.get(key);

frestr = key;

}

}

map.remove(frestr);

list.add(frestr);

}

**return** list;

}

/\*\*

\* 关闭输入输入流

\*

\* **@param** inout

\*/

**private** **void** close(Closeable inout) {

**if** (inout != **null**) {

**try** {

inout.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

}

/\*\*

\* 提示 将红楼梦文本文件拆分为120个章节的方法

\*

\* **@param** content

\* **@return** 返回120个元素的字符串数组

\*/

**private** String[] splitContentToChapter(String content) {

// 提示 使用 content.split(" 第[一,二,三,四,五,六,七,八,九,十,零]{1,5}回 ");正则表达拆分

// 百度一下正则表达式

String contents[] = content.split(" 第[一,二,三,四,五,六,七,八,九,十,零]{1,5}回 ");

**return** contents;

}

/\*\*

\* 统计红楼梦章节字符串str出现的频率

\* **@param** str

\* **@return**

\* **@throws** Exception

\*/

**public** **int**[] getStringFrequent(String str) **throws** Exception {

**int** count [] = **new** **int** [120];

// 分章节统计

**for**(**int** i = 0; i < 120; i++) {

count[i] = 0;

}

String f = "C:";

Analysis ana = **new** Analysis (f);

String totaltxt = "";

totaltxt = ana.readFromTxt(filename);

String [] contents = ana.splitContentToChapter(totaltxt);

**int** strlen = str.length();

**for**(**int** i = 0; i < 120; i++) {

**int** contentlen = contents[i+1].length();

**char** ch[] = contents[i+1].toCharArray();

StringBuffer []splitstb = **new** StringBuffer [contentlen + 1 - strlen];

String []splitstr = **new** String [contentlen + 1 - strlen];

// 构造出所有与统计目标str同样长度的词

**for**(**int** j = 0; j < contentlen + 1 - strlen; j++) {

splitstb[j] = **new** StringBuffer("");

**for**(**int** u = j; u < j + strlen; u++) {

splitstb[j].append(ch[u]);

}

splitstr[j] = splitstb[j].toString();

// 与str比较 如果相同 即str出现次数+1

**if** (str.equals(splitstr[j])) {

count[i] ++;

}

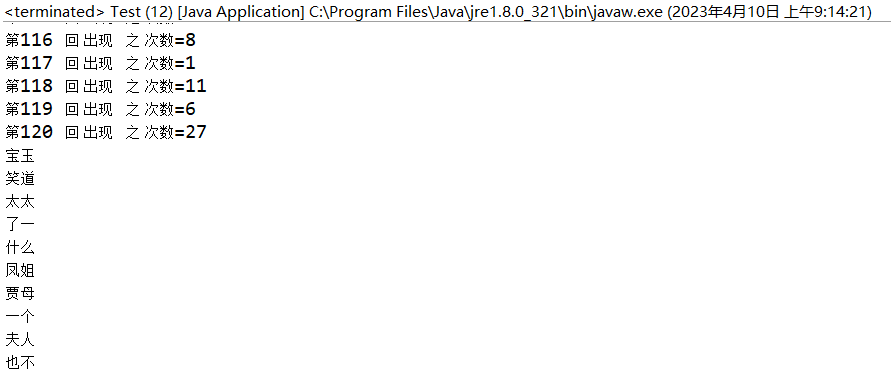
}

}

**return** count;

}

1. **实验结果**



1. **实验中遇到的问题及解决方法**

在这个实验中，我学会了如何读入文件，以及如何对读入的一长串字符串做分割的处理。我还巩固了对map的理解，练习了对value值的写入与覆盖。

问题：出现非汉语的词汇 即词汇中含标点等符号

解决：预先在文本中提出标点等符号，再做组词的操作