

**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 王杨森**

**学 号 3020244116**

**年 级 2020级**

**班 级 软件工程3班**

**时 间 2022年 4月7日**

1. **实验目的**
2. 熟悉Eclipse/MyEclipse；
3. 熟悉Java中String/StringBuffer/数组的使用；
4. 熟悉文件的读写；
5. **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 {

String text = readFromTxt(filename);//文件读入

text = text.replaceAll("[\\pP\t\r]", "");//利用正则表达式将标点符号、换号符等删除

chapters = splitContentToChapter(text);//要操作的数组

}

/\*\*

\* 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**[] counts = **new** **int**[120];

**for**(**int** i = 1; i < chapters.length; i++) {

**int** cnt = 0;

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

**for**(**int** j = 0; j < chapters[i].length() - len + 1; j++) {

String temp = chapters[i].substring(j, j + len);

**if**(temp.equals(str)) {//利用遍历在每一章中搜索能够匹配的str

cnt++;

}

}

counts[i - 1] = cnt;

}

**return** counts;

}

用一个HashMap存放每个单词及其出现次数，用一个List存放map键值对，按照value进行升序排序，前十个元素的key即出现频率最高的N个词。

/\*\*

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

\* **@param** n

\* **@return**

\*/

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

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

Map<String, Integer>map = **new** HashMap<>();

**for**(**int** i = 1; i < chapters.length; i++) {

String text = chapters[i];

**for**(**int** j = 0; j < text.length() - 1; j++){

**if**(text.charAt(j) == ' ' || text.charAt(j + 1) == ' ') **continue**;

String word = text.substring(j, j + 2);

**if**(!map.containsKey(word)) {

map.put(word, 1);

} **else** {

map.put(word, map.get(word) + 1);

}

}

}

List<Map.Entry<String, Integer>> sortList = **new** ArrayList<>(map.entrySet());

sortList.sort((o1, o2) -> o2.getValue().compareTo(o1.getValue()));

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

res.add(sortList.get(i).getKey());

}

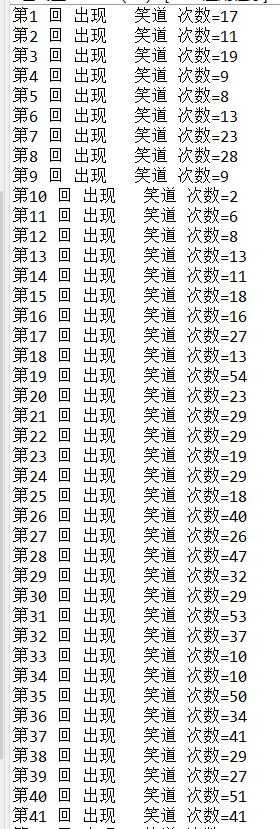
**return** res;

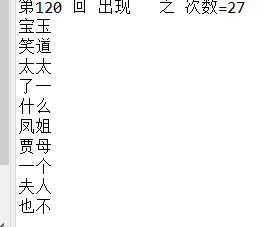
}

1. **实验结果**

编写测试用例并进行验证：

 程序运行结果如下图所示：





1. **实验中遇到的问题及解决方法**

在这次实验中，我加深了对于String的使用技巧的理解，并且学习了对于List数据结构的应用。同时也复习了Java文件读写的使用方法。

在统计每章出现的特定str时，第一个问题是要将文本进行分割，这里就需要用到正则表达式，按照特定的字符如第一回进行分割，然后在每一章中，进行遍历查询即可。

在统计出现次数最多的前十名时，一开始没有意识到文本中可能会存在大量标点符号，经过查询资料，\pP在正则表达式中意为标点符号，那么在构造函数时，就可以把所有的标点符号和换行符替换为空字符。然后对全部文本进行遍历，每两个字作为一个字符串存入到一个List中，由于需要得到出现次数最多的前十个词，那么仅仅存入字符串是不够的，需要同时存入出现次数，所以就需要一个HashMap，用字符串作为key，出现次数作为value，将这个键值对存入一个List，再按照value的升序对这个List进行排序，那么这个List的前十个元素的key即是要统计的答案。