

Java 课程上机报告

上机题目：Java 的输入输出

张俊华 16030199025 (组长)

李金鑫 16030199026

李天浩 16030199027

上机时间：2017/05/06 14:00-18:00

地点：EIII-203

一、 小组名单

学号	姓名	工作
16030199025	张俊华	完成程序、练习例程、调试、小组讨论
16030199026	李金鑫	完成程序、练习例程、调试、小组讨论
16030199027	李天浩	完成程序、练习例程、调试、小组讨论

二、 题目

实验目标：

熟悉 Java 的泛型；了解 Java 的泛型特点；初步掌握 Java 的泛型编程方法。

实验要求：

掌握 Java 输入输出类；掌握 Java 输入输出特点；掌握 Java 输入输出编程方法。

实验要求：

编写一个程序，程序实现对用户指定的文本文件中的英文字符和字符串的个数进行统计的功能，并将结果根据用户选择输出至结果文件或屏幕。

（1）构建统计类，该类实现对 I/O 的操纵；实现对文本文件中英文字符、字符串的统计；实现对统计结果的输出。

（2）构建测试类，该类实现与用户的交互，向用户提示操作信息，并接收用户的操作请求。

程序应具有良好的人机交互性能，即：程序应向用户提示功能说明，并可根据用户的功能选择，执行对应的功能，并给出带详细描述信息的最终执行结果。

三、 题目分析：

本实验要求编写一个程序，实现以下功能：

1. 对用户指定文本文件中的英文字符进行统计。
2. 对文本文件中的字符串数进行统计
3. 可选择统计结果输出到屏幕或文件

1. 对指定文本文件中的英文字符进行统计：

要实现对英文字符的统计首先要创建一个 File 对象，打开用户指定的文件。然后创建 FileReader 流，从 FileReader 流读取字符，为了提高效率，选择使用 BufferedReader 流与 FileReader 对接，调用 BufferedReader 的 read 方法读取字符。遍历整个文件，将每个字符出现的频率保存在数组中，即可完成对英文字符数量的统计

2. 对文本文件中的字符串数进行统计

与统计字符类似，调用 bufferedReader 的 readline 方法，每次读取一行，本实验定义字符串数为单词数，即统计文件中的单词总数。调用 String 的 split 方法，将读到的内容按空格符分隔成字符串数组，数组大小即为文本的单词数。

3. 可选择统计结果输出到屏幕或文件

构建 Action 类，Action 类存在字段 outputToFile，用来保存用户设置，当 outputToFile 值为 true 时，调用输出方式为文件的方法，否则调用输出方式为控制台的方法。

四、程序实现：

张俊华：

1. 实验环境：

IntelliJ IDEA 2017.1.2

Build #IU-171.4249.39, built on April 25, 2017

JRE: 1.8.0_112-release-736-b16 amd64

JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

Windows 10 10.0

2. 实现过程：

在开始编写程序前先对程序的整体结构和各个类之间的关系进行设计。定义三个类：

Counter：对文件进行读取，实现统计字符数和字符串数

Actions：程序功能和逻辑的实现

UserInterface：与用户交互，包括控制台的输入，输出到控制台或文件

在 Counter 中编写了以下方法：

```
public class Counter {  
    /**  
     *统计file 中的英文字符数  
     *@param file 目标文件对象缓存流  
     *@return 字符数目数组  
     *@author 张俊华 16030199025  
     */  
    public Integer[] countCharacter(File file) throws IOException {  
        Integer[] letterFrequency = new Integer[26]; //数组保存每个字母出现个  
        for (int i = 0; i < 26; i++) {  
            letterFrequency[i] = 0;  
        }  
        int tempChar;  
        FileReader fileReader = new FileReader(file);  
        BufferedReader bufferedReader = new BufferedReader(fileReader);  
        while ((tempChar = bufferedReader.read()) != -1) {  
            数
```

```

        if (tempChar >= 'a' && tempChar <= 'z') {
            letterFrequency[tempChar - 'a']++;
        }
        if (tempChar >= 'A' && tempChar <= 'Z') {
            letterFrequency[tempChar - 'A']++;
        }
    }
    bufferedReader.close();
    fileReader.close();
    return letterFrequency;
}

/**
 *统计file 中的单词数
 *@param file 目标文件对象
 *@return 字符串数
 *@author 张俊华 16030199025
 */
public Integer countAllString(File file) throws IOException {

    Integer number = 0;
    FileReader fileReader = new FileReader(file);
    BufferedReader bufferedReader = new BufferedReader(fileReader);
    String tempString;
    while ((tempString = bufferedReader.readLine()) != null) {
        number += tempString.split(" ").length;
    }
    bufferedReader.close();
    fileReader.close();
    return number;
}

```

定义 Actions 类，在 Actions 中，定义私有字段：

```

private File inputFile; //输入文件
private File outputFile; //输出文件
private boolean outputToFile = false; //输出到文件设置

```

通过调用 Counter 类中的方法实现字符或字符串的统计。例如

```

/**
 *实现统计用户文件中的字符串数量并输出
 *@author 张俊华 16030199025
 */
public void countString() throws IOException {

    if (outputFile != null && outputToFile) {
        outputer.outputStringNumber(counter.countAllString(inputFile),
outputFile);
    } else {
        outputer.outputStringNumber(counter.countAllString(inputFile));
    }
}

```

定义包 `userInterface` 其中含有三个类 `Inputer`、`Outputer`、`Prompt`，分别实现输入，输出和选项提示操作。

其中 `Outputer` 包含输出到控制台和文件的方法，`Actions` 调用输出到文件的方法时，需要传入参数 `outputFile`，通过 `bufferedWriter.write` 方法实现文本的写入，例如

```
/**
 * 输出文本文件中的英文字符串个数到文件
 * @param stringNumber 字符串个数
 * @param file 目标文本文件
 * @author 张俊华 16030199025
 */
public void outputStringNumber(Integer stringNumber, File file) throws
IOException {
    FileWriter fileWriter = new FileWriter(file);
    BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
    System.out.println("文件信息统计成功! ,统计结果正在输出到文件中...");
    System.out.println("本文件共有" + stringNumber + "个单词");
    bufferedWriter.write("本文件共有" + stringNumber + "个单词\r\n");
    bufferedWriter.close();
    fileWriter.close();
}
```

最后 `Actions` 中的 `UserChoise` 方法通过调用类中的其他方法，执行用户操作

```
/**
 * 执行用户想执行的功能
 * @return boolean 是否进行了选择
 * @author 张俊华 16030199025
 */
public boolean UserChoise() {
    prompt.ShowMenu();
    ShowSettings();
    //打印菜单
    String choose = inputer.ScanChoise();
    try {
        if (choose.equals("s")) {
            Sets();
        } else if (choose.equals("1")) {
            countChar();
        } else if (choose.equals("2")) {
            countString();
        } else if (choose.equals("3")) {
            countTheString();
        } else if (choose.equals("q")) {
            return false;
        }
    } catch (NullPointerException np) {
        System.out.println(np.getMessage());
        System.out.println("未定义文件， 请进行设置! ");
        Sets();
    } catch (IOException io) {
        prompt.TipWrong();
        System.out.println(io.getMessage());
    }
}
```

```
System.out.println("请输入任意字符继续...");  
new Scanner(System.in).next();  
  
return true;  
}
```

运行结果:

```
+-----+  
请输入选项: s  
+-----+  
输入文档: 未定义  
输出文档: 未定义  
当前输出方式: 控制台  
+-----+  
+-----+  
| 选项      功能      |  
| 1         修改输入文件 |  
| 2         修改输出文件 |  
| 3         修改输出方式 |  
+-----+  
请输入选项: 1  
请输入文件名: a.txt  
请输入任意字符继续...
```

```
+-----+  
请输入选项: s  
+-----+  
输入文档: a.txt  
输出文档: 未定义  
当前输出方式: 控制台  
+-----+  
+-----+  
| 选项      功能      |  
| 1         修改输入文件 |  
| 2         修改输出文件 |  
| 3         修改输出方式 |  
+-----+  
请输入选项: 2  
请输入文件名: b.txt  
请输入任意字符继续...
```

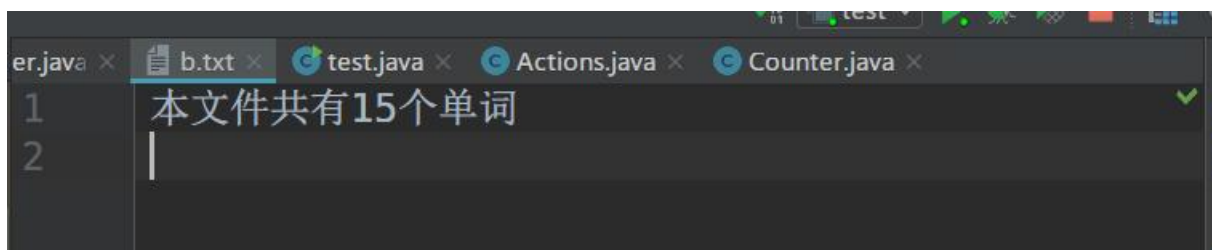
```
+-----+
|  选项          功能  |
|  1             统计字符  |
|  2             统计单词  |
|  3             统计指定单词  |
|  s             系统设置  |
|  q             退出系统  |
+-----+
+-----+
输入文档: a.txt
输出文档: b.txt
当前输出方式: 控制台
+-----+
请输入选项: 1
文件信息统计成功!
本文件共有96个字符
文本文件中各字符出现次数如下:
a      8次
b      1次
c      1次
d      5次
e      10次
f      12次
g      0次
h      7次
i      7次
j      8次
k      4次
l      3次
m      1次
n      1次
o      2次
p      0次
```

```
+-----+
+-----+
|  选项          功能  |
|  1             修改输入文件  |
|  2             修改输出文件  |
|  3             修改输出方式  |
+-----+
请输入选项: 3
当前输出方式: 控制台
修改吗 (y/n) :
请输入选项: y
修改成功! 当前输出方式: 文件+控制台
请输入任意字符继续...
```

```

+-----+
| 选项      功能      |
| 1          统计字符  |
| 2          统计单词  |
| 3          统计指定单词 |
| s          系统设置  |
| q          退出系统  |
+-----+
+-----+
输入文档: a.txt
输出文档: b.txt
当前输出方式: 文件+控制台
+-----+
请输入选项: 2
文件信息统计成功! ,统计结果正在输出到文件中...
本文件共有15个单词
请输入任意字符继续...

```



李天浩:

1. 实验环境:

IntelliJ IDEA 2017.1.2

Build #IU-171.4249.39, built on April 25, 2017

JRE: 1.8.0_112-release-736-b16 amd64

JVM: OpenJDK 64-Bit Server VM by JetBrains s.r.o

Windows 10 10.0

2. 实现过程

首先设计了 Container 类实现程序的主要功能, 包括从文件读入, 统计字母出现次数, 字符串出现次数, 然后是 Main 类实现交互功能, 并且作为程序的入口.

统计字母出现次数使用了一个数组, 统计字符串出现次数使用了一个
HashMap<String, Integer>

设计了构造器用于从传入的 FileName 处读入
下面的方法分别实现的对应功能

```
import java.io.*;
import java.util.*;

public class Container {
    private File myFile;
    private HashMap<String, Integer> StringCounter;
    private int[] CharacterCounter;
    StringBuffer text=new StringBuffer();
    Container(String FileName) {
        BufferedReader in = null;
        try {
            myFile = new File(FileName);
            in = new BufferedReader(new FileReader(myFile));
            CharacterCounter = new int[260];
            StringCounter = new HashMap<>();
            String line;
            while((line=in.readLine()) != null) {
                line.toLowerCase();
                text.append(line+' ');
            }
        }
        catch (IOException e) {
            System.out.println("文件读取错误");
            System.exit(1);
        }
        finally{
            try{
                if(in!=null)
                    in.close();
            }
            catch(IOException e) {
                System.out.println("文件关闭错误");
                System.exit(1);
            }
        }
    }

    public void CountChar() {
        for (int i = 0; i < text.length(); i++) {
            char c;
            c = text.charAt(i);
            CharacterCounter[(int)c]++;
        }
        for(int i=0;i<=25;i++){
```

```

        System.out.print((char) (i+'a'));
        System.out.print('\t');
        System.out.println(CharacterCounter[i+'a']);
    }
}

public void CountString() {
    String[] words=text.toString().split(" ");
    StringBuffer ans=new StringBuffer();
    for (int i = 0; i < words.length; i++) {
        if (!StringCounter.containsKey(words[i])) StringCounter.put(words[i], 1);
        else StringCounter.put(words[i],StringCounter.get(words[i])+1 );
    }
    Iterator iterator = StringCounter.keySet().iterator();
    while(iterator.hasNext()) {
        String word = (String) iterator.next();
        ans.append("单词: ").append(word).append(" 次数
").append(StringCounter.get(word)).append("\n");
    }
    System.out.println(ans);
}
}
}

```

运行结果

测试文件

```

This General Public License does not permit incorporating your program into
proprietary programs.  If your program is a subroutine library, you may
consider it more useful to permit linking proprietary applications with the
library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.

```

a	17
b	5
c	8
d	4
e	25
f	4
g	5
h	7
i	28
j	0
k	1
l	9
m	7
n	16
o	22
p	12
q	0
r	28
s	18
t	20
u	11
v	0
w	3
x	0
y	9
z	0

单词: 次数2
单词: do, 次数1
单词: use 次数1
单词: License 次数2
单词: your 次数2
单词: program 次数2
单词: subroutine 次数1
单词: library, 次数1
单词: library. 次数1
单词: GNU 次数1
单词: into 次数1
单词: not 次数1
单词: Lesser 次数1
单词: proprietary 次数2
单词: does 次数1
单词: of 次数1
单词: License. 次数1
单词: This 次数1
单词: consider 次数1
单词: programs. 次数1
单词: you 次数2
单词: incorporating 次数1
单词: a 次数1
单词: may 次数1
单词: more 次数1
单词: linking 次数1
单词: want 次数1
单词: this 次数2
单词: is 次数2
单词: Public 次数2
单词: it 次数1
单词: instead 次数1
单词: the 次数2
单词: with 次数1
单词: what 次数1
单词: permit 次数2
单词: General 次数2
单词: to 次数2
单词: If 次数2
单词: useful 次数1
单词: applications 次数1

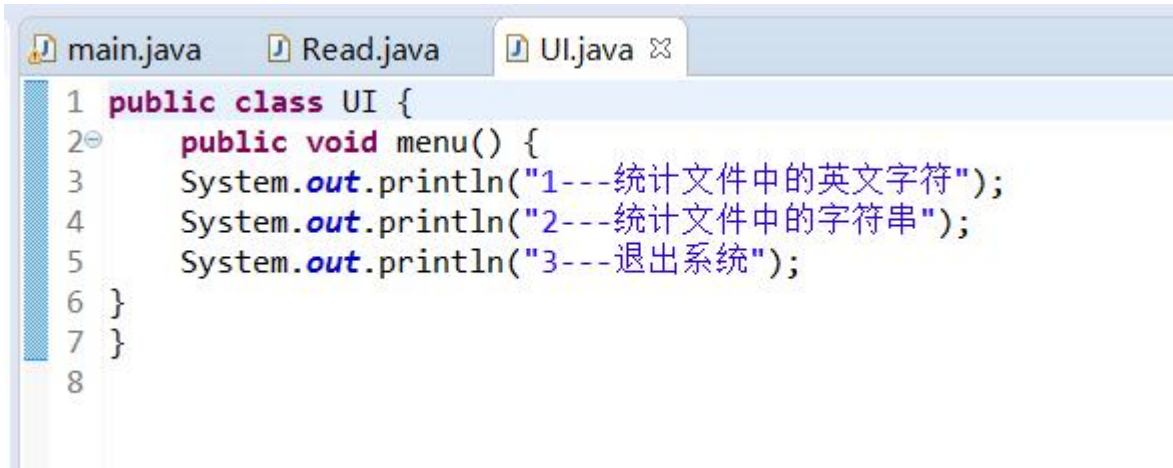
李金鑫：

实现过程：

定义三个类：main(用于用户交互，选择统计的类型以及输出方式)

UI(界面)

Read(从文件中读取数据并且进行统计、输出)



```
1 public class UI {  
2     public void menu() {  
3         System.out.println("1---统计文件中的英文字符");  
4         System.out.println("2---统计文件中的字符串");  
5         System.out.println("3---退出系统");  
6     }  
7 }  
8
```

```

main.java Read.java UI.java
1 import java.util.*;
2
3 public class main {
4     public static void main(String[] args) throws IOException{
5         Scanner in=new Scanner(System.in);
6         UI p=new UI();
7         p.menu();
8         int n=in.nextInt();
9         Read r=new Read();
10        r.re();
11        while (n!=3){
12            switch(n){
13                case 1:
14                    int n1;
15                    System.out.println("1---将结果输出到屏幕上");
16                    System.out.println("2---将结果输出到文件中");
17                    n1=in.nextInt();
18                    if (n1==1) r.output1();else r.wr(1);
19                    break;
20                case 2:
21                    int n2;
22                    System.out.println("1---将结果输出到屏幕上");
23                    System.out.println("2---将结果输出到文件中");
24                    n1=in.nextInt();
25                    if (n1==1) r.output2();else r.wr(2);
26                    break;
27            }
28            n=in.nextInt();
29        }
30    }
31 }

```

可以看到每种功能给了两种输出方法，分别对应. output 方法和. wr 方法。

```

main.java Read.java UI.java
1 import java.io.*;
2 public class Read {
3     int[] wd=new int[30];
4     int sum=0;
5     public void re() throws FileNotFoundException, IOException {
6         FileReader fr=new FileReader ("D:\\1.txt");
7         BufferedReader bufr=new BufferedReader(fr);
8         String str=null;
9         while ((str=bufr.readLine())!=null){
10             sum++;
11             int len=str.length();
12             for (int i=0;i<len;i++){
13                 char c=str.charAt(i);
14                 c=Character.toLowerCase(c);
15                 if ((c>='a')&&(c<='z')) wd[c-'a']++;
16                 if (c==' ') sum++;
17             }
18         }
19         bufr.close();
20     }
}

```

用 `BufferedReader` 读取 1.txt 中的内容，每次按行读取，把读取的内容放在 `str` 字符串中，然后按位读取本行的数据，先用 `str.charAt` 方法把要读取的单个字符拿出来，然后用 `toLowerCase` 方法设为小写，方便我们判断它是否为一个字母。用 `wd` 数组存每个字母出现的次数，例如 `w[0]` 表示 `a` 出现的次数，以此类推。

同时，我们判断字符串的个数可以用一个小技巧，每次读到一个换行或者一个空格都是多了一个字符串，利用这个特性，用 `sum` 存储字符串的个数。

当读完 1.txt 中的数据关闭流。

```

1 public void output1(){
2     for (int i=0;i<=25;i++){
3         System.out.println((char)(i+97)+"出现次数为"+wd[i]);
4     }
5 }
6 public void output2(){
7     System.out.println("字符串有"+(sum-1)+"个");
8 }

```

输出到屏幕上的操作用 `System.out.println` 即可。

```


29 public void wr(int x) throws IOException{
30     FileWriter fw=new FileWriter("D:\\2.txt");
31     BufferedWriter bufw=new BufferedWriter(fw);
32     if (x==1) {
33         for (int i=0;i<=25;i++)
34             bufw.write((char)(i+97)+"出现次数为"+wd[i)+"\r\n");
35     }else bufw.write("字符串有"+(sum-1)+"个");
36     bufw.flush();
37     bufw.close();
38 }
39 }
40

```

输出到文件中我们用 `BufferedWriter`，根据传进来的参数 `x` 选择输出的数据。最后要关闭缓冲区和流。

运行结果:

第一个功能的实现:

 1.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
qwrqw tgeqwt tgerwtg
yrj rqw ADD

main (3) [Java Application] C:\Program Files\Java\jre1.8.0_112\bin\javaw.exe (2)

1---统计文件中的英文字母

2---统计文件中的字符串

3---退出系统

1

1---将结果输出到屏幕上

2---将结果输出到文件中

1

a出现次数为1

b出现次数为0

c出现次数为0

d出现次数为2

e出现次数为2

f出现次数为0

g出现次数为3

h出现次数为0

i出现次数为0

j出现次数为1

k出现次数为0

l出现次数为0

m出现次数为0

n出现次数为0

o出现次数为0

p出现次数为0

q出现次数为4

r出现次数为4

s出现次数为0

t出现次数为4

u出现次数为0

v出现次数为0

w出现次数为5

x出现次数为0

y出现次数为1

z出现次数为0

1

1---将结果输出到屏幕上

2---将结果输出到文件中

2

<

a出现次数为1
b出现次数为0
c出现次数为0
d出现次数为2
e出现次数为2
f出现次数为0
g出现次数为3
h出现次数为0
i出现次数为0
j出现次数为1
k出现次数为0
l出现次数为0
m出现次数为0
n出现次数为0
o出现次数为0
p出现次数为0
q出现次数为4
r出现次数为4
s出现次数为0
t出现次数为4
u出现次数为0
v出现次数为0
w出现次数为5
x出现次数为0
y出现次数为1
z出现次数为0

第二个功能的实现：



```
main (3) [Java Application] C:\Program Files\Java\jre1.8.0_112\bin\javaw.e
1---统计文件中的英文字符
2---统计文件中的字符串
3---退出系统
2
1---将结果输出到屏幕上
2---将结果输出到文件中
1
字符串有6个
2
1---将结果输出到屏幕上
2---将结果输出到文件中
2
```

2.txt - 记事本

文件(E) 编辑(E) 格式(O) 查看(V) 帮助(H)

字符串有6个

二. 小组讨论内容:

张俊华:

小组内李金鑫同学程序, 各类分工明确, 运行结果满足要求, 不足之处在于, 统计字符串个数时, 采用空格作为字符串标志, 导致有连续空格时会出现统计字符串数多于实际字符串数的情况, 而且无法完成题目要求的统计指定字符串的数量。

小组内李天浩同学的程序, 设计了构造器读入文件, 使用 HashMap 存储字符串数据, 程序设计巧妙, 考虑到了程序可能出现的异常情况, 很好的满足了题目要求

李金鑫:

张俊华同学的程序很好的完成了题目要求, 交互页面精美, 考虑全面, 可读性强。

李天浩同学的程序也完成了要求, 并且用了 HashMap 存储字符串, 但是我认为最后输出的单词出现多少次并不是十分符合题目要求。

李天浩：

我认为单纯使用空格和回车来计数是不科学的,连续空格或者回车断开了单词的情况也需要考虑.看了小组内另外两位同学的代码实现,发现自己在 UI 设计方面还存在很多不足,要多多向他们学习.

六、个人总结：

张俊华：

通过这次上机,我对 java 的文件读写操作有了较为深刻的认识,了解到 java 的输入输出流,以及流之间的对接操作等,同时也学习到了 Java 异常处理,Java 严格的异常处理机制能够准确的找出出错点,捕获并处理异常使得程序逻辑大大简化,同时也使得程序的稳定性得以提高。

本次上机的不足之处在于异常处理的类型较少,没有对所有可能出现的异常都进行捕获并处理,程序还有完善的空间

李金鑫：

通过这次上机,我对 java 的文件都写有了较为深刻的认识,学会了运用 **BufferedWriter** 和 **BufferedReader** 这种带有默认缓冲的字符输出输入流,并且认识了 java 的异常处理。

同时对于读取方式有了初步的认识,本题中按行读取加计算空格能方便我们统计字符串的个数。

本次上机也暴露出了一些问题,我对 java 文件操作运用并不熟练,对异常处理的理解也不够深刻。

李天浩：

通过这次上机,我对 java 的文件读写有了一些自己的认识,学会了使用多种方法读入文件,这次选用的是 **BufferedReader**. 也进一步了解了 java 的异常处理机制.通过对异常处理,减少了程序运行不稳定的发生.

本次上机也暴露出来了很多问题,比如对知识掌握,运用的不熟练,对异常处理的类型较少.