

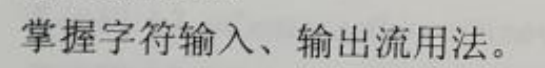
Java程序设计实验报告

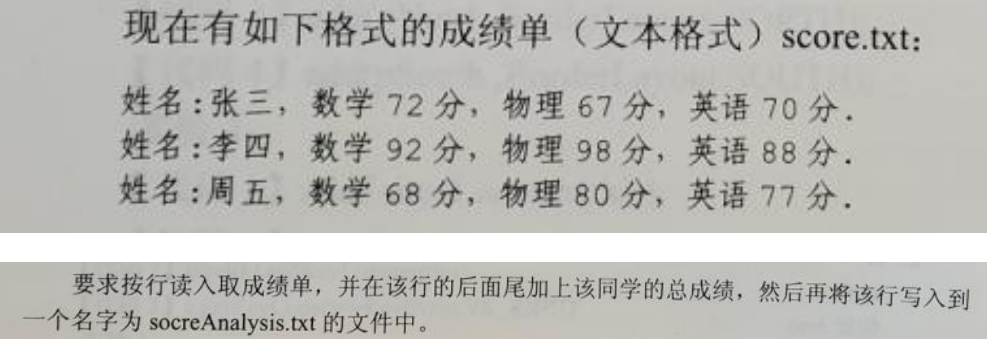
实验报告

【实验题目】 Java实验7

#### 【实验内容】

##### 1. 练习1

**实验目的**

**实验要求**

###### 代码：

**Total.java**

package Score;

import java.util.InputMismatchException;

import java.util.Scanner;

public class Total {

    public static double getTotal(String s) {// 计算总分的函数

        Scanner sc = new Scanner(s);

        sc.useDelimiter("[^0123456789.]+");// 设置分隔符

        double total = 0;// 总分

        while (sc.hasNext()) {//获取字符串直到为null

            try {

                double score = sc.nextDouble();

                total += score;

            } catch (InputMismatchException exp) {// 获取输入不匹配的错误

                String t = sc.next();

            }

        }

        return total;

    }

}

**Test1.java**

package Score;

import java.io.\*;

public class Test1 {

    public static void main(String args[]) {

        File fRead = new File("score.txt");

        File fWrite = new File("scoreAnalysis.txt");

        try {

            Reader in = new FileReader(fRead); // 创建指向fRead的in流

            Writer out = new FileWriter(fWrite); // 以尾加方式创建指向文件fWrite的out流

            BufferedWriter bw = new BufferedWriter(out); // 创建指向out的bufferedWrite流

            BufferedReader br = new BufferedReader(in); // 创建指向in的bufferRead流

            String line = br.readLine();// 获取第一行

            int i = 0;

            while (line != null) {

                bw.write(line);// 写入第一行

                bw.write("总分:");// 行结尾写上总分

                bw.write(String.valueOf(Total.getTotal(line)));// 写上字符串型的分数

                bw.write("分.");

                bw.newLine();// 换行

                line = br.readLine();//读取下一行

            }

            br.close();

            bw.close();

        } catch (IOException e) {

            System.out.println(e.toString());

        }

    }

}

###### 输出结果：

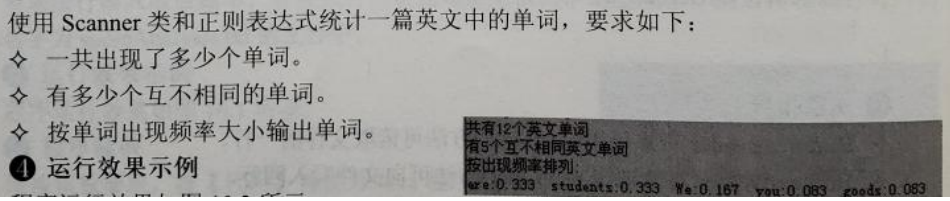
###### 

###### 分析：

这个程序就是用先用readLine读取txt文件的内容，根据指定字符进行分割，之后算出总分，即三个数字相加，最后将原来读出的先写入新的txt文件，之后写入总分即可。

##### 2.练习2

**实验目的**

**实验要求**

###### 代码：

**Word.java**

package Word;

import java.io.File;

import java.io.IOException;

import java.util.Scanner;

import java.util.Vector;

public class Words {

    Scanner sc = null;

    String regex;

    Vector<String> word, diff;

    File file = new File("word.txt");

    Words() {

        word = new Vector<String>();

        diff = new Vector<String>();

        regex = "[\\s\\d\\p{Punct}]+";//由空格、数字和符号(!"#$%&'()\*+,-./:;<=>?@[\]^\_`{|}~)组成的正则表达式

        try {

            sc = new Scanner(file);

            sc.useDelimiter(regex);//分割字符串

        } catch (IOException exp) {

            System.out.println(exp.toString());

        }

    }

    void setName(String name) {//设置文件名字

        file = new File(name);

        try {

            sc = new Scanner(file);

            sc.useDelimiter(regex);

        } catch (IOException e) {

            System.out.println(e.toString());

        }

    }

    public Vector<String> getAllWord() {//获得全部元素

        return word;

    }

    public Vector<String> getNoSameWord() {//不同元素

        return diff;

    }

    public void wordStatistic() {//获得不同元素

        try {

            while (sc.hasNext()) {

                String str = sc.next();

                word.add(str);

                if (!diff.contains(str))//不包含

                    diff.add(str);//就加入其中

            }

        } catch (Exception e) {

        }

    }

}

**Test2.java**

package Word;

import java.util.Vector;

public class Test2 {

    public static void main(String args[]) {

        Vector<String> word, diff;

        Words w = new Words();

        w.setName("word.txt");//设置文件名

        w.wordStatistic();//调用WordStatistic()方法

        word = w.getAllWord();//获得全部文字

        diff = w.getNoSameWord();//不同文字

        System.out.println("共有" + word.size() + "个英文单词");

        System.out.println("有" + diff.size() + "个互不相同英文单词");

        System.out.println("按出现频率排列：");

        int count[] = new int[diff.size()];//创建不同种类个数大小的数组

        for (int i = 0; i < diff.size(); i++) {

            String s1 = diff.elementAt(i);//获得i索引处的元素

            for (int j = 0; j < word.size(); j++) {

                String s2 = word.elementAt(j);//获得j索引处的元素

                if (s1.equals(s2))//比较二者是否一样

                    count[i]++;//计数

            }

        }

        for (int m = 0; m < diff.size(); m++) {

            for (int n = m + 1; n < diff.size(); n++) {

                if (count[n] > count[m]) {//排序

                    String temp = diff.elementAt(m);

                    diff.setElementAt(diff.elementAt(n), m);//获得索引处设置给定元素

                    diff.setElementAt(temp, n);

                    int t = count[m];//交换顺序

                    count[m] = count[n];

                    count[n] = t;

                }

            }

        }

        for (int m = 0; m < diff.size(); m++) {

            double frequency = (1.0 \* count[m]) / word.size();//计算百分占比

            System.out.printf("%s：%-7.3f", diff.elementAt(m), frequency);

        }

    }

}

###### 输出结果：

###### 

###### 分析：

先将字符串分解成每个单独的英文字符，统计单词个数，之后进行逐个比较就可以得到不同单词的个数，之后就是排序算法，按出现次数由大到小进行排序，最后就是算出占比即可。

##### 练习3

**实验目的**熟练使用RandomAccessFile流

**实验要求**

###### 代码：

import java.io.\*;

public class Reverse {

    public static void main(String[] args) {

        RandomAccessFile inAndOut = null;

        try {

            inAndOut = new RandomAccessFile("text1.txt", "r");

            long num = inAndOut.length() / 3;// 长度计算

            String str = null;

            StringBuilder sb = new StringBuilder();// new一个可变的字符序列

            while (--num != 0) {// 逐个添加到序列

                if ((str = inAndOut.readLine()) != null) {// 不打印空字符

                    sb.append(str);

                    sb.append("\n");

                }

            }

            inAndOut.close();

            System.out.println(sb.reverse());// 可变字符串倒叙输出

        } catch (IOException e) {

            System.out.println(e.toString());

        }

    }

}

###### 输出结果：

测试数据：

结果：

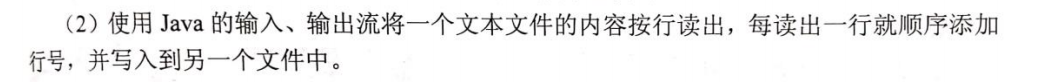


###### 分析：

先求长度，之后将，每一个字符传入可变字符串中进行记录，之后直接reverse就得到倒叙的了。

##### 练习4

**实验目的**熟练使用Java的输入输出流

**实验要求**

###### 代码：

import java.io.\*;

import java.util.ArrayList;

public class File {

    public static void main(String[] args) {

        try {

            BufferedReader br = new BufferedReader(new FileReader("text2.txt"));

            BufferedWriter bw = new BufferedWriter(new FileWriter("text2\_\_ans.txt"));

            int count = 0;// 记录行数

            String str = null;

            ArrayList<String> list = new ArrayList<>();// 定义列表

            while ((str = br.readLine()) != null) {

                list.add(++count + "." + str);// 每次向列表中存入一个字符串

            }

            br.close();

            for (int i = 0; i < list.size(); i++) {// 将新的字符串写入新文件

                bw.write(list.get(i));// 写入列表对应下标的一行

                bw.newLine();// 新一行

                bw.flush();

            }

            bw.close();

        } catch (IOException e) {

            System.out.println(e.toString());

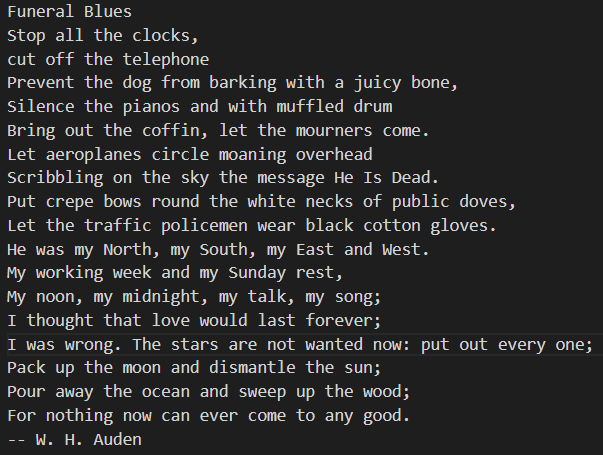
        }

    }

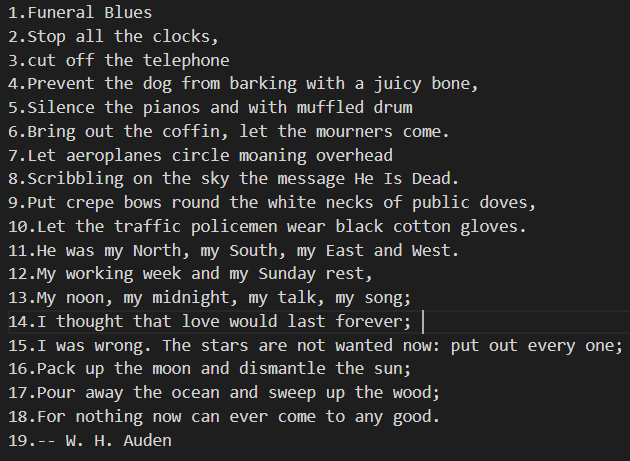
}

###### 输出结果：

测试数据：



结果：



###### 分析：

从原文件中每读出一行存入到列表一行，并且存入前加上序号，之后就是将列表按下标进行写入到新的文件中就可以啦。

#### 【小结与讨论】

1. 因为输入的字符串长度是不可限定的，为了能存入分割出来的字符串，就不能用静态数组进行存储，所以我们就会想到两个工具，第一个就是可变字符串，第二个就是列表，这些都是可变大小的。
2. 在读写文件的时候，直接写文件名.txt时必须保证txt文件和我们的程序.java文件在一个文件夹内，否则就得用绝对路径（即D://\*\*\*）进行读取，否则就会找不到指定文件。