

# Java 第七章 输入输出上机练习

22009200601 汤栋文

2023 年 5 月 20 日

## 目录

1. 个人信息.....	2
2. 任务重述.....	2
3. PPT 练习 .....	2
3.1 RenameFile on P7.....	2
3.2 FileCopy on P30.....	2
3.3 DataIO on P36.....	3
3.4 StandardIO on P40 .....	3
3.5 RandomAccessTest on P49 .....	3
3.6 (Un)SerializeDate on P53 .....	3
4. 编写统计字符程序 .....	4
4.0 整体说明.....	4
4.1 统计段落数/单词数/字母数 .....	4
4.2 按给定字符匹配查找.....	5
4.3 在允许的条件下复制文件 .....	5
5. 总结.....	5

## 1. 个人信息

姓名：汤栋文

学号：22009200601

日期：2023 年 5 月 20 日

## 2. 任务重述

1. 练习 PPT 中的全部小练习，尝试对小练习中各部分进行修改，并观察修改后的执行效果。
2. 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。
  - a) 针对给定（英文）文档，统计文档的段落数、单词数及每个字母出现的次数。
  - b) 针对一个文件夹（文件夹下全部文件及子文件夹）下全部文件，根据给定数据串，检索出全部包含该字符串的全部文件，并将结果列表。
  - c) 针对给定源文件名及目的文件名（文件名以 main 函数参数方式给定），实现将源文件拷贝至目的文件。若源文件名与目的文件名相同、目的文件已存在、源文件不存在。不执行拷贝，并将信息反馈给用户。
  - d) 程序应具有良好的人机交互性能。

## 3. PPT 练习

### 3.1 RenameFile on P7

File 类对象实例是一个在内存中描述文件信息的结构，或者使描述路径信息的结构，而不是文件或者目录本身，在创建这个 File 实例的时候并不会把文件读进来。文件是否存在均不会对 File 类对象实例整体产生影响。

File 类对象实例中存储了很多的文件相关的信息。想要获取文件信息时调用起来非常方便。（在这里需要注意一个小问题，有时候文件明明就在那，但是不可读又不可写，出现一些奇怪的问题，经常是由于文件权限导致的，这是一个比较隐蔽的错误不太容易发现。）

### 3.2 FileCopy on P30

这里的复制并不是我们传统意义上的复制，而是把一个文件中的信息作为输入流，把输出流指向到另一个文件，从而实现把一个文件中的内容以流的形式读进来，再以流的形式输出。一种非常朴素非常底层的复制方式。

仔细一想，其实我们直接复制粘贴文件的时候，计算机的底层操作方式，也就是把一个文件全部读出来，然后再全部写到另一个地方去。在我没有思考这个问题之前，一直觉得复制文件是一个很简单的东西，被没有多么复杂的操作，现在看来并不是这样。

### 3.3 DataIO on P36

在数据输出流中按照特定的顺序输出出去，再按照原来的顺序读回来。我比较在意的是，这些数据具体是怎么存的，也就是说，如果我直接用记事本打开会看到什么东西。



可以看到，记事本会直接用 UTF-8 的编码格式来读取这个文件，所以按照 UTF-8 存储的部分仍然可读，而其他形式存储的数据，都会被按照 UTF-8 方式读成乱码。

### 3.4 StandardIO on P40

顾名思义标准输入输出，从输入流中读取输入。不过我没能搞清楚那个异常处理在干什么，我没能以任何方式触发这个异常，大概是输入流不存在或者其他问题的时候才会触发的吧（？）

### 3.5 RandomAccessTest on P49

RandomAccessFile 类可以进行文件的随机访问。感觉就像是控制一个光标，将光标移动到指定位置（通过 seek 方法），然后对光标处的文件内容进行操作。

但是问题有一个，我发现中文会发生乱码，这还是编码格式的问题，不知道有没有特定的方法可以按照特定的编码格式读取文件，或者读进来以后重新解码。

### 3.6 (Un)SerializeDate on P53

这种存储方式可以很方便地把一整个对象实例一下子全部存下来，下次读取的时候再一下子全部读进来，可以说是非常方便了。

但是这里面有一个坑，这是图书馆管理系统中我使用地存储方式，就是说我们在存储的时候最好使用 Serializable 接口，并且设定一个 serialVersionUID 否则在下次读入的时候因为这个版本标识不一样就会导致报错读不进来。

```
public class Database implements Serializable{

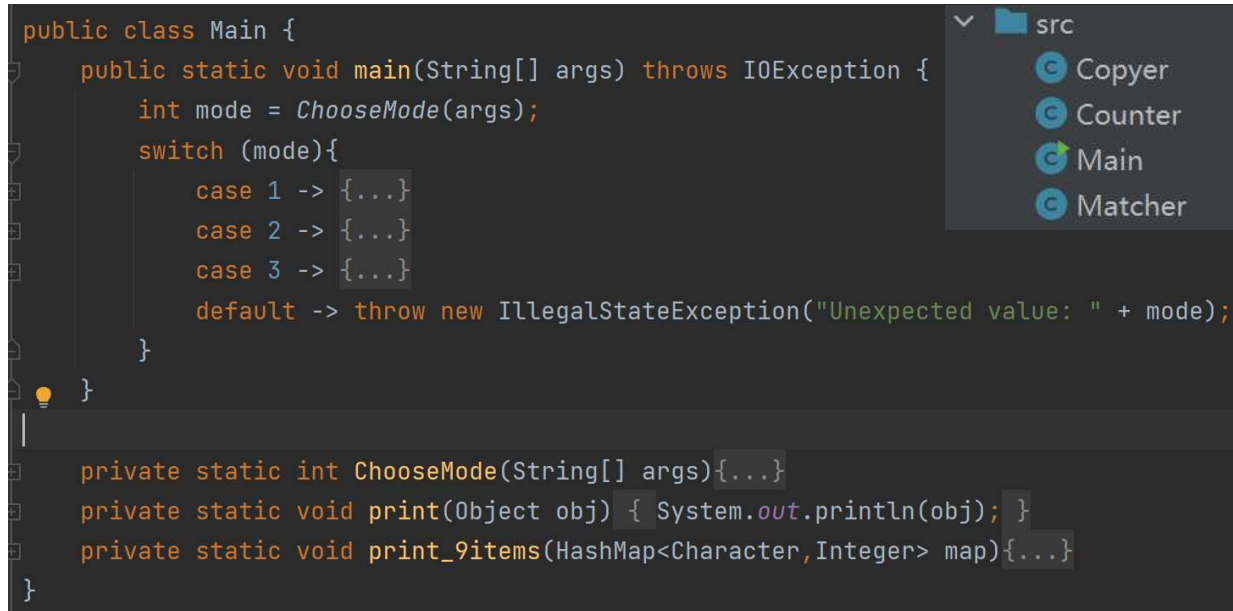
    @Serial private static final long serialVersionUID = 1024L;

    public Database() { return; } // create a new database
    public Database(String path) throws IOException, ClassNotFoundException {
        File file = new File(path); if (!file.exists()) return;
        FileInputStream infile = new FileInputStream(file);
        ObjectInputStream instream = new ObjectInputStream(infile);
        Database database_old = (Database) instream.readObject();
    }
}
```

## 4. 编写统计字符程序

### 4.0 整体说明

所有的参数使用命令行输入。Counter, Matcher, Copyer 分别用于三个任务。  
Main 函数分析输入类型, 分配给指定的类处理。



```

public class Main {
    public static void main(String[] args) throws IOException {
        int mode = ChooseMode(args);
        switch (mode){
            case 1 -> {...}
            case 2 -> {...}
            case 3 -> {...}
            default -> throw new IllegalStateException("Unexpected value: " + mode);
        }
    }

    private static int ChooseMode(String[] args){...}
    private static void print(Object obj) { System.out.println(obj); }
    private static void print_9items(HashMap<Character,Integer> map){...}
}

```

Project Explorer (src):

- Copyer
- Counter
- Main
- Matcher

### 4.1 统计段落数/单词数/字母数

段落统计时计算进行了几次 `readLine` 即可, 其中当读出的字符串为空时说明有空行, 不记录。

```

int count = 0;
while( (str=in.readLine()) !=null ){ if(!str.isEmpty()) count++; }

```

单词统计时, 利用正则表达式筛选掉所有标点, 连续空格替换为单个空格, 然后计数空格。

```

while( (str=in.readLine()) !=null ){
    str = str.replaceAll( regex: "[^0-9a-zA-Z ]", replacement: "");
    str = str.replaceAll( regex: "+", replacement: " ");
    str = str.trim();
    String[] strlist = str.split( regex: " ");
    count += strlist.length;
}

```

字符计数时, 先进性类似单词统计的预处理, 然后遍历整个文档, 利用 Map 计数。

```

for (Character character: str.toCharArray()){
    if (character==' ') continue;
    count.merge(character, value: 1, Integer::sum);
}

```

(仅粘贴了代码核心部分, 具体参考代码)

## 4.2 按给定字符匹配查找

递归获得给定目录下的所有可读的文件。存到一个 ArrayList 里面。

```
public ArrayList<File> findfiles(File file){
    ArrayList<File> result = new ArrayList<>();
    assert file.isDirectory();
    File[] files = file.listFiles();
    if (files == null) return result;
    for (File this_file: files) {
        if (this_file.isFile()) result.add(this_file);
        else if (this_file.isDirectory()) result.addAll(findfiles(this_file));
    } return result;
}
```

然后遍历每个文件，如果存在了相同的字符串就把它打印出来。

```
for (File this_file: files){
    BufferedReader bufferin = new BufferedReader(new FileReader(this_file));
    if(matcher.contains(args[1], bufferin)) print(this_file);
}
```

(仅粘贴了代码核心部分，具体参考代码)

## 4.3 在允许的条件下复制文件

先判断是否可以复制，多出口返回可复制情况。

```
public String copyable(File src, File dst){
    if (Objects.equals(src.toString(), dst.toString())) return "Duplicate names";
    if (dst.exists()) return "Destination exists";
    if (!src.exists()) return "Source not exists";
    return "Copyable";
}
```

利用文件输入输出流复制文件。

```
public void copy(String src, String dst) throws IOException {
    FileInputStream in = new FileInputStream(src);
    FileOutputStream out = new FileOutputStream(dst);
    int content;
    while( (content=in.read())!=-1 )
        out.write(content);
}
```

(仅粘贴了代码核心部分，具体参考代码)

## 5. 总结

革命尚未成功，同志仍需努力