**每日作业卷答案**

**JavaSE第11天**

传智播客.黑马程序员

# 关卡1

## 训练案例1

### 训练描述

利用字节输出流一次写一个字节的方式向C盘的a.txt文件输出内容。

### 操作步骤答案

**public** **class** Test01\_01 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建字节输出流FileOutputStream对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("c:/a.txt");

// 2.调用字节输出流的write(int byte)方法写出数据

fos.write(97);

// 3.关闭流

fos.close();

}

}

## 训练案例2

### 训练描述

利用字节输出流一次写一个字节数组的方式向C盘的b.txt文件输出内容。

### 操作步骤答案

**public** **class** Test01\_02 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建字节输出流FileOutputStream对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("c:/b.txt");

// 2.调用字节输出流的write(byte[] buf)方法写出数据。

**byte**[] buf = "i love java".getBytes();

fos.write(buf);

// 3.关闭资源

fos.close();

}

}

## 训练案例3

### 训练描述

利用字节输出流对象往C盘下c.txt文件输出5句：”i love java”

要求：

1. 不能覆盖文件中原有的内容
2. 每一句输出各占一行

### 操作步骤答案

**public** **class** Test01\_03 {

**public** **static** **void** main(String[] args) **throws** IOException{

// 1.创建字节输出流FileOutputStream对象并指定文件路径。以追加方式

FileOutputStream fos = **new** FileOutputStream("c:/c.txt",**true**);

// 2.调用字节输出流的write方法写出数据

// 2.1 要输出的字符串

String content = "i love java \r\n";

**for** (**int** i = 0; i < 5; i++) {

fos.write(content.getBytes());

}

// 3.关闭流

fos.close();

}

}

## 训练案例4

### 训练描述

利用字节输入流读取C盘文件a.txt的内容，使用循环读取，一次读取一个字节，直到读取到文件末尾。将读取的字节输出到控制台

### 操作步骤答案

**public** **class** Test01\_04 {

**public** **static** **void** main(String[] args) **throws** IOException{

// 创建字节输入流对象并关联文件

FileInputStream fis = **new** FileInputStream("c:/a.txt");

// 定义变量接收读取的字节

**int** len = -1;

// 循环从流中读取数据

**while**((len = fis.read()) != -1) {

System.***out***.print((**char**)len);

}

// 关闭流

fis.close();

}

}

## 训练案例5

### 训练描述

利用字节输入流读取C盘文件b.txt的内容，使用循环读取，一次读取一个字节数组，直到读取到文件末尾，将读取到的字节数组转换成字符串输出到控制台。

### 操作步骤答案

**public** **class** Test01\_05 {

**public** **static** **void** main(String[] args) **throws** IOException{

// 创建字节输入流对象并关联文件

FileInputStream fis = **new** FileInputStream("c:/b.txt");

// 定义字节数组存放读取的字节数

**byte**[] buffer = **new** **byte**[1024];

// 定义变量接收读取的字节

**int** len = -1;

// 循环从流中读取数据

**while**((len = fis.read(buffer)) != -1) {

System.***out***.print(**new** String(buffer,0,len));

}

// 关闭流

fis.close();

}

}

## 训练案例6

### 训练描述

利用字节流将C盘下的a.png图片复制到D盘下(文件名保存一致)

要求：

1.一次读写一个字节的方式

### 操作步骤答案

**public** **class** Test01\_06 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 创建字节输入流对象并关联文件

FileInputStream fis = **new** FileInputStream("c:/a.png");

// 创建字节输出流对象并关联文件

FileOutputStream fos = **new** FileOutputStream("d:/a.png");

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = fis.read()) != -1) {

// 每读取一个字节的数据就写出到目标文件中

fos.write(len);

}

// 关闭流

fis.close();

fos.close();

}

}

## 训练案例7

### 训练描述

用字节流将C盘下的a.png图片复制到D盘下(文件名保存一致)

要求：一次读写一个字节数组的方式进行复制

### 操作步骤答案

**public** **class** Test01\_07 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 创建字节输入流对象并关联文件

FileInputStream fis = **new** FileInputStream("c:/a.png");

// 创建字节输出流对象并关联文件

FileOutputStream fos = **new** FileOutputStream("d:/a.png");

// 定义数据接收读取的字节数

**byte**[] buffer = **new** **byte**[1024];

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = fis.read(buffer)) != -1) {

// 将字节数组中的数据写出到目标文件中

fos.write(buffer,0,len);

}

// 关闭流

fis.close();

fos.close();

}

}

## 训练案例8

### 训练描述

利用高效字节输出流往C盘下的d.txt文件输出一个字节数。

### 操作步骤答案

**public** **class** Test01\_08 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 创建字节输出流FileOutputStream对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("c:/d.txt");

// 利用字节输出流创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(fos);

// 调用高效字节输出流对象的write(int byte)方法写出一个字节数据

bos.write(97);

// 关闭流

bos.close();

}

}

## 训练案例9

### 训练描述

利用高效字节输出流往C盘下的e.txt文件写出一个字节数组数据，如写出：”i love java”

### 操作步骤答案

**public** **class** Test01\_09 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 创建字节输出流FileOutputStream对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("c:/e.txt");

// 利用字节输出流创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(fos);

// 调用高效字节输出流对象的write(byte[] buff)方法写出一个字节数据

bos.write("i love java".getBytes());

// 关闭流

bos.close();

}

}

## 训练案例10

### 训练描述

利用高效字节输入流和高效字节输出流完成文件的复制。

要求：

1. 将C盘下的b.png文件复制到D盘下

一次读写一个字节方式复制

### 操作步骤答案

**public** **class** Test01\_10 {

**public** **static** **void** main(String[] args) **throws** IOException{

// 创建字节输入流对象并关联文件路径

FileInputStream fis = **new** FileInputStream("c:/b.png");

// 利用字节输出流对象创建高效字节输出流对象

BufferedInputStream bis = **new** BufferedInputStream(fis);

// 创建字节输出流对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("d:/b.png");

// 利用字节输出流创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(fos);

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = bis.read()) != -1) {

// 每读取一个字节的数据就写出到目标文件中

bos.write(len);

}

// 关闭流

bis.close();

bos.close();

}

}

## 训练案例11

### 训练描述

利用高效字节输入流和高效字节输出流完成文件的复制。

要求：

1. 将C盘下的c.png文件复制到D盘下
2. 一次读写一个字节数组方式复制

### 操作步骤答案

**public** **class** Test01\_10 {

**public** **static** **void** main(String[] args) **throws** IOException{

// 创建字节输入流对象并关联文件路径

FileInputStream fis = **new** FileInputStream("c:/c.png");

// 利用字节输出流对象创建高效字节输出流对象

BufferedInputStream bis = **new** BufferedInputStream(fis);

// 创建字节输出流对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("d:/c.png");

// 利用字节输出流创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(fos);

// 定义字节数组接收读取的字节

**byte**[] buffer = **new** **byte**[1024];

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = bis.read(buffer)) != -1) {

// 每读取一个字节的数据就写出到目标文件中

bos.write(buffer,0,len);

}

// 关闭流

bis.close();

bos.close();

}

}

## 训练案例12

### 训练描述

四种复制文件方式比较

1. 字节流复制文件一次读写一个字节
2. 字节流复制文件一次读写一个字节数组
3. 高效流复制文件一次读写一个字节
4. 高效流复制文件一次读取一个字节数组

利用上面四种方式复制同一个文件，输出每一种复制方式花费的时间。

### 操作步骤答案

**public** **class** Test01\_10 {

**public** **static** **void** main(String[] args) **throws** IOException{

System.***out***.println("字节流复制文件一次读写一个字节耗时："+ *copy01*());

System.***out***.println("字节流复制文件一次读写一个字节数组耗时："+ *copy02*());

System.***out***.println("高效流复制文件一次读写一个字节耗时："+ *copy03*());

System.***out***.println("高效流复制文件一次读取一个字节数组耗时："+ *copy04*());

}

// 字节流复制文件一次读写一个字节

**public** **static** **long** copy01() **throws** IOException{

// 获得当前时间毫秒值

**long** startTime = System.*currentTimeMillis*();

// 创建字节输入流对象并关联文件

FileInputStream fis = **new** FileInputStream("aaa.jpg");

// 创建字节输出流对象并关联文件

FileOutputStream fos = **new** FileOutputStream("ttt.jpg");

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = fis.read()) != -1) {

// 每读取一个字节的数据就写出到目标文件中

fos.write(len);

}

// 关闭流

fis.close();

fos.close();

**return** System.*currentTimeMillis*() - startTime;

}

// 字节流复制文件一次读写一个字节数组

**public** **static** **long** copy02() **throws** IOException{

// 获得当前时间毫秒值

**long** startTime = System.*currentTimeMillis*();

// 创建字节输入流对象并关联文件

FileInputStream fis = **new** FileInputStream("aaa.jpg");

// 创建字节输出流对象并关联文件

FileOutputStream fos = **new** FileOutputStream("eee.jpg");

// 定义数据接收读取的字节数

**byte**[] buffer = **new** **byte**[1024];

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = fis.read(buffer)) != -1) {

// 将字节数组中的数据写出到目标文件中

fos.write(buffer,0,len);

}

// 关闭流

fis.close();

fos.close();

**return** System.*currentTimeMillis*() - startTime;

}

// 高效流复制文件一次读写一个字节

**public** **static** **long** copy03() **throws** IOException{

// 获得当前时间毫秒值

**long** startTime = System.*currentTimeMillis*();

// 创建字节输入流对象并关联文件路径

FileInputStream fis = **new** FileInputStream("aaa.jpg");

// 利用字节输出流对象创建高效字节输出流对象

BufferedInputStream bis = **new** BufferedInputStream(fis);

// 创建字节输出流对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("jjj.jpg");

// 利用字节输出流创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(fos);

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = bis.read()) != -1) {

// 每读取一个字节的数据就写出到目标文件中

bos.write(len);

}

// 关闭流

bis.close();

bos.close();

**return** System.*currentTimeMillis*() - startTime;

}

// 高效流复制文件一次读取一个字节数组

**public** **static** **long** copy04() **throws** IOException{

// 获得当前时间毫秒值

**long** startTime = System.*currentTimeMillis*();

// 创建字节输入流对象并关联文件路径

FileInputStream fis = **new** FileInputStream("aaa.jpg");

// 利用字节输出流对象创建高效字节输出流对象

BufferedInputStream bis = **new** BufferedInputStream(fis);

// 创建字节输出流对象并指定文件路径。

FileOutputStream fos = **new** FileOutputStream("ddd.jpg");

// 利用字节输出流创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(fos);

// 定义字节数组接收读取的字节

**byte**[] buffer = **new** **byte**[1024];

// 定义变量接收读取的字节数

**int** len = -1;

// 循环读取图片数据

**while**((len = bis.read(buffer)) != -1) {

// 每读取一个字节的数据就写出到目标文件中

bos.write(buffer,0,len);

}

// 关闭流

bis.close();

bos.close();

**return** System.*currentTimeMillis*() - startTime;

}

}

# 关卡2

## 训练案例1

### 训练描述

从控制台循环接收用户录入的学生信息，输入格式为：学号-学生名字

将学生信息保存到C盘下面的stu.txt文件中，一个学生信息占据一行数据。

当用户输入end时停止输入。

\* 要求使用字节输出流写出数据

### 操作步骤答案

**public** **class** Test02\_01 {

**public** **static** **void** main(String[] args) {

**try** {

// 创建字节输出流对象

FileOutputStream fos = **new** FileOutputStream("c:/stu.txt");

// 创建键盘录入对象

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入学生信息，格式：学号-姓名");

// 使用死循环不停的接收用户输入的学生数据

**while**(**true**) {

// 接收用户输入的学生数据

String line = sc.nextLine();

// 判断输入的内容是否是end，是则终止循环

**if**(line.equals("end")){

**break**;

}

// 将数据写出到文件中。

fos.write(line.getBytes());

// 写出换行符号

fos.write("\n".getBytes());

}

// 释放资源

fos.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

## 训练案例2

### 训练描述

使用高效流字符输入读取上一题保存学生信息的文件，每次读取一行信息。

每读取到一行信息就创建一个学生对象并添加到集合中。

遍历集合打印学生信息到控制台。

### 操作步骤答案

**public** **class** Test02\_02 {

**public** **static** **void** main(String[] args) {

// 创建高效流字符流对象

BufferedReader br = **new** BufferedReader(**new** FileReader("c:/stu.txt"));

// 定义集合存放学生对象

List<Student> stus = **new** ArrayList<Student>();

// 定义字符串接收读取的数据

String line = **null**;

// 使用循环从文件中读取数据

**while**((line = br.readLine()) != **null**) {

// 使用-分隔字符串

String[] strs = line.split("-");

// 创建学生对象

Student stu = **new** Student(strs[0],strs[1]);

// 将学生添加到集合中

stus.add(stu);

}

// 关流释放资源

br.close();

// 遍历集合打印学生信息

**for** (Student student : stus) {

System.***out***.println(student.getId()+"="+student.getName());

}

}

## 训练案例3

### 训练描述

在C盘下创建一个目录d1,在目录d1下创建创建若干个文本文件，并在每一个文本文件中添加若干内容。利用字节高效流将该目录下的所有文件复制到C盘下的d2目录下。

### 操作步骤答案

**public** **class** Test02\_03 {

**public** **static** **void** main(String[] args) {

// 创建文件对象

File dir = **new** File("c:/d1");

// 获得该文件下的所有文件

File[] files = dir.listFiles();

// 创建目标文件

File destDir = **new** File("c:/d2");

// 遍历文件数组

**for** (File file : files) {

// 创建高效字节输入流对象

BufferedInputStream bis = **new** BufferedInputStream(**new** FileInputStream(file));

// 创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(**new** FileOutputStream(**new** File(destDir,file.getName())));

// 定义字节数组接收读取的字节数

**byte**[] buffer = **new** **byte**[1024];

// 定义变量接收实际读取的字节数

**int** len = -1;

// 从输入流中循环读取数据

**while**((len = bis.read(buffer)) != -1) {

// 利用输入流将数据写出到目标文件中

bos.write(buffer, 0, len);

}

// 关闭资源

bis.close();

bos.close();

}

}

}

## 训练案例4

### 训练描述

在C盘下创建一个目录d1,在目录d1下创建若干个txt文件，并在每一个txt文件中添加若干内容。将所有txt文件的内容复制到D盘下的copy.txt中。

\* 要求所有txt的文件内容都复制到同一个文件中

### 操作步骤答案

**public** **class** Test02\_04 {

**public** **static** **void** main(String[] args) {

// 创建文件对象

File dir = **new** File("c:/d1");

// 获得该文件下的所有文件

File[] files = dir.listFiles();

// 创建目标文件

File destFile = **new** File("d:/copy.txt");

// 遍历文件数组

**for** (File file : files) {

// 创建高效字节输入流对象

BufferedInputStream bis = **new** BufferedInputStream(**new** FileInputStream(file));

// 创建高效字节输出流对象

BufferedOutputStream bos = **new** BufferedOutputStream(**new** FileOutputStream(destFile,**true**));

// 定义字节数组接收读取的字节数

**byte**[] buffer = **new** **byte**[1024];

// 定义变量接收实际读取的字节数

**int** len = -1;

// 从输入流中循环读取数据

**while**((len = bis.read(buffer)) != -1) {

// 利用输入流将数据写出到目标文件中

bos.write(buffer, 0, len);

}

// 关闭资源

bis.close();

bos.close();

}

}

}

## 训练案例5

### 训练描述

在C盘下有一个文本文件test.txt(里面的内容由数字和字母组成)

定义一个方法统计test.txt文件中指定字符出现的次数。

比如a字符在文件中出现了10次则调用方法传入a后，方法内部输出：a出现10次

### 操作步骤答案

**public** **class** Test02\_05 {

**public** **static** **void** main(String[] args) {

// 调用方法

*calcuteCount*('a');

}

/\*

\* 统计字符在文件中出现的次数

\*/

**public** **static** **void** calcuteCount(**char** ch) {

FileInputStream fis = **null**;

**try** {

// 定义一个计数变量，统计字符出现的次数

**int** count = 0;

// 创建字节输入流

fis = **new** FileInputStream("test.txt");

// 循环读取数据

**int** len = -1;

**while** ((len = fis.read()) != -1) {

// 将读取的字节转换成字符

**char** c = (**char**) len;

// 判断字符c和外界传入的字符是否相同

**if** (c == ch) {

count ++;

}

}

System.***out***.println(ch+"出现了" + count +"次");

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**if** (fis != **null**)

**try** {

fis.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

}

## 训练案例6

### 训练描述

使用集合存储10个1-50(含50)的随机偶数元素,要求数字不能重复,按指定格式输出到C盘的num.txt中,例如: 48,44,40,38,34,30,26......

### 操作步骤答案

**public** **class** Test02\_06 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 1.创建集合对象HashSet

HashSet<Integer> set = **new** HashSet<Integer>();

// 2.创建随机数对象Random

Random r = **new** Random();

// 3.使用循环随机生成10个1-50(含50)的偶数

**while**(set.size() < 10) {

// 4.随机生成一个整数

**int** num = r.nextInt(50) + 1;

// 5.判断是否偶数

**if**(num % 2 == 0) {

// 6.将生成的偶数添加到集合中

set.add(num);

}

}

// 将集中的元素按指定格式拼接成字符串

StringBuilder sb = **new** StringBuilder();

**for** (Integer num : set) {

sb.append(num).append(",");

}

// 删除最后一个分号

sb.deleteCharAt(sb.length() - 1);

// 创建高效流关联目标文件，

BufferedOutputStream bos = **new** BufferedOutputStream(**new** FileOutputStream("c:/num.txt"));

bos.write(sb.toString().getBytes());

// 关闭流释放资源

bos.close();

}

}

# 关卡3

## 训练案例1

### 训练描述

模拟用户登录和注册功能实现。

接收用户输入的用户名和密码，然后和文件中存储的用户名、密码匹配。

如果文件已经存在用户名并且密码也是正确的，提示登录成功，否则提示登录失败。

如果文件中不存在该用户名，则使用该用户名和密码注册一个新的账号，并提示注册成功。

### 操作步骤答案

**public** **class** Test03\_01 {

**public** **static** **void** main(String[] args) {

// 创建键盘流入对象

Scanner sc = **new** Scanner(System.***in***);

// 请输入用户名和密码

System.***out***.println("请输入用户名：");

String username = sc.nextLine();

System.***out***.println("请输入密码：");

String password = sc.nextLine();

// 调用登录方法

*login*(username,password);

}

/\*

\* 登录方法：判断用户名和密码是否正确，正确返回true,否则返回false

\*/

**public** **static** **boolean** login(String username,String password){

// 声明高效字符输入流

BufferedReader br = **null**;

**try** {

// 创建文件对象

File file = **new** File("user.txt");

// 判断文件是否存在

**if**(!file.exists()) {

// 创建文件

file.createNewFile();

// 登录失败

**return** **false**;

}

// 创建高效字符输入流

br = **new** BufferedReader(**new** FileReader(file));

// 定义变量接收读取的每一行数据

String line = **null**;

// 定义标记记录用户名是否存在

**boolean** exist = **false**;

// 循环读取每一行数据

**while**((line = br.readLine())!= **null**) {

// 使用-分隔

String[] strs = line.split("=");

// 判断用户名是否正确

**if**(strs[0].equals(username)) {

exist = **true**;

**if**(strs[1].equals(password)) {

System.***out***.println("登录成功");

**return** **true**;

} **else** {

System.***out***.println("密码错误，登录失败");

**return** **false**;

}

}

}

**if**(!exist) { // 用户名不存在

System.***out***.println("用户名不存在");

// 注册账号

*register*(username,password);

}

} **catch** (Exception e) {

e.printStackTrace();

} **finally** {

// 关闭流并释放资源

**try** {

**if**(br != **null**) br.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

**return** **false**;

}

/\*

\* 注册用户方法

\*/

**public** **static** **void** register(String username,String password) **throws** IOException{

// 创建字符输出流对象

BufferedWriter bw = **new** BufferedWriter(**new** FileWriter("user.txt",**true**));

// 保存用户名和I密码

bw.write(username);

bw.write("=");

bw.write(password);

// 写换行符

bw.newLine();

// 关闭流

bw.close();

}

}

## 训练案例2

### 训练描述

在d盘目录下有一个加密文件a.txt（文件里只有英文和数字），密码是“heima”

当密码输入正确时才能读取文件里的数据。现要求用代码来模拟读取文件的过程，并统计文件里各个字母出现的次数，并把统计结果按照如下格式输出到d盘的count.txt中。

a：2个

b：3个

c：4个

............

### 操作步骤答案

**public** **class** Test03\_02 {

**public** **static** **void** main(String[] args) **throws** IOException {

// 创建键盘流入对象

Scanner sc = **new** Scanner(System.***in***);

// 请输入密码

System.***out***.println("请输入密码：");

// 判断密码是否正确

**while**(**true**){

String password = sc.nextLine();

**if**(password.equals("heima")) {

**break**;

} **else** {

System.***out***.println("密码错误，请重新输入：");

}

}

// 创建map集合存放数据

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

// 统计每个字符出现的次数

*calCuteCharCount*(map);

// 将统计结果保存到文件中

*saveResult*(map);

}

/\*

\* 将统计结果保存到文件中

\*/

**public** **static** **void** saveResult(Map<Character,Integer> map) **throws** IOException {

// 创建字符输出流对象

BufferedWriter bw = **new** BufferedWriter(**new** FileWriter("result.txt",**true**));

// 遍历map集合

**for**(Entry<Character,Integer> entry:map.entrySet()) {

// a：2个

bw.write(entry.getKey()+":" + entry.getValue()+"个");

// 写换行符

bw.newLine();

}

// 关闭流

bw.close();

}

/\*

\* 统计每个字符出现的次数

\*/

**public** **static** **void** calCuteCharCount(Map<Character,Integer> map) **throws** IOException {

// 创建字输入符高效流

BufferedReader br = **new** BufferedReader(**new** FileReader("c.txt"));

String line = **null**;

// 循环读取数据

**while**((line = br.readLine()) != **null**) {

// 遍历字符串

**for** (**int** index = 0; index < line.length(); index++) {

// 获得每一个字符

Character ch = line.charAt(index);

// 判断map集合是否有该字符对应的键

**if**(map.containsKey(ch)) {

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

} **else** {

map.put(ch, 1);

}

}

}

// 关闭流

br.close();

}

}

## 训练案例3

### 训练描述

从控制台获取输入的文件目录然后将该目录(包含子目录)下的.java文件复制到D:/java文件夹中,并统计java文件的个数.

提示:如果有相同的名称的文件,如果两个Test01.java,则拷贝到目标文件夹时只能有一个Test01.java,另一个Test01.java要修改为另一个名称:该名称可随机生成,只要不重复即可.

### 操作步骤答案

**public** **class** Test03\_03 {

// 定义统计变量记录Java文件的总数

**static** **int** *count* = 0;

**public** **static** **void** main(String[] args) **throws** IOException{

// 获得源文件夹

File srcDir = *inputFile*();

// 创建目标文件夹对象

File destDir = **new** File("d:/java");

destDir.mkdirs();

// 拷贝Java文件

*copyJavaFile*(srcDir, destDir);

System.***out***.println("count = " + *count*);

}

/\*

\* 将srcDir文件夹下的所有.Java文件拷贝到destDir文件夹下

\*/

**public** **static** **void** copyJavaFile(File srcDir,File destDir) **throws** IOException{

// 获得源文件夹下所有的.java文件和子文件夹

File[] files = srcDir.listFiles(**new** FileFilter() {

@Override

**public** **boolean** accept(File pathname) {

**if**(pathname.isDirectory() || pathname.getName().endsWith(".java"))**return** **true**;

**return** **false**;

}

});

// 遍历文件数组

**for** (File file : files) {

// 判断file是否是文件夹

**if**(file.isDirectory()) {

// 递归调用当前方法

*copyJavaFile*(file, destDir);

} **else** { // 如果是文件,拷贝到目标文件夹destDir下

// 获得要拷贝的文件名

String fileName = file.getName();

// 在目标文件夹destDir下创建一个同名文件名

File destFile = **new** File(destDir,fileName);

// 判断目标文件是否存在

**while**(destFile.exists()) {

// 重命名目标文件名

destFile = **new** File(destDir,**new** Random().nextInt(100000) + fileName);

}

System.***out***.println(destFile);

*copyFile*(file, destFile);

}

}

}

/\*

\* 字节流缓冲区流读写字节数组

\*/

**private** **static** **void** copyFile(File srcFile, File destFile) **throws** IOException {

// 计数器加一

*count* ++;

// 创建字节缓冲输入流

BufferedInputStream bis = **new** BufferedInputStream(**new** FileInputStream(srcFile));

// 创建字节缓冲输出流

BufferedOutputStream bos = **new** BufferedOutputStream(**new** FileOutputStream(destFile));

**int** len = 0;

**byte**[] buffer = **new** **byte**[1024];

// 循环读取数据

**while**((len = bis.read(buffer)) !=-1) {

// 利用输出流往输出目的写数据

bos.write(buffer,0,len);

}

// 关闭资源

bis.close();

bos.close();

}

/\*

\* 根据用户输入的路径字符串创建文件对象

\*/

**private** **static** File inputFile() {

// 创建键盘输入对象

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("请输入一个文件夹路径：");

// 接收用户输入的路径

String filePath = sc.nextLine();

// 根据路径字符串创建文件对象

File dir = **new** File(filePath);

**if**(!dir.exists()) **throw** **new** RuntimeException("你输入的文件路径不存在！！！");

// 判断是否是文件夹

**if**(!dir.isDirectory()){

**throw** **new** RuntimeException("输入的路径不是文件夹路径！！！");

}

**return** dir;

}

}