**中软国际考题-笔试**

1. 单选题

1、File类型中定义了什么方法来判断一个文件是否存在？（B）

A.isNone B.exists C. createNewFile D.haveFile

2、下面的程序段的功能是 （D）

File file1=new File("e:\\xxx\\yyy");

file1.mkdir();

A. 在当前目录下生成子目录：\xxx\yyy  B. 在当前目录下生成文件xxx.yyy

C. 生成目录：e:\xxx\yyy  D. 以上说法都不对

3.下列流中那个使用了缓冲区技术（A）

A．BufferedOutputStream B．FileInputStream

C．DataOutputStream D．FileReader

4.与InputStream流相对应的Java系统的标准输入对象是（A）

A．System.in B．System.out C．System.err D．System.exit()

5、以下通过哪个类对象可以实现从文件读取数据（D）

A.FileWriter B.FileOutputStream C.StringWriter D.FileInputStream

6、关于 Java 流的叙述错误的是（ D ）

A. 流是 Java 语言处理 I/O 的方式 B. 从概念上讲，流就是一系列的字节序列或字符序列

C. 引入流的概念的目的是使得在处理不同的数据输入输出时编程更加方便

D. 流是Java惟一的非面向对象的语言构成

7、为了从文本文件中逐行读取内容，应该使用哪个处理流对象？（ A ）

A．BufferedReader B．BufferedWriter

C．BufferedInputStream D．BufferedOutputStream

8、使用Java IO流实现对文本文件的读写过程中，需要处理下列哪个异常。（ B ）

A. ClassNotFoundException

B. IOException

C. SQLException

D. RemoteException

9、用“new FileOutputStream(“data.txt”,true)”创建一个 FileOutputStream 实例对象，则下面（B ）说法是正确的。

A、如果文件“data.txt”存在，则将抛出 IOException 异常

B、如果文件“data.txt”存在，则将在文件的末尾开始添加新内容

C、如果文件“data.txt”存在，则将覆盖掉文件中已有的内容

D、如果文件“data.txt”不存在，则将抛出 IOException 异常

10、读取一个序列化的对象一般使用哪种Stream（D）

A、InputStream B、FileReader C、DataInputStream D、ObjectStream

11、从InputStream对象中如何创建一个Reader对象？（C）

A.使用InputStream类中定义的createReader()方法

B.用Reader类中的createReader()方法

C.构造一个InputStreamReader实例，将InputStream对象作为InputStreamReader类构造器的参数传入

D.构造一个OutputStreamReader实例，将InputStream对象作为OutputStreamReader类构造器 的参数传入

1. 多选题
2. 能指定字符编码的I/O流类型是？（B、D）
3. Reader B. InputStreamReader C. BufferedReader
4. OutputStreamWriter E. Writer
5. PrintWriter
6. File对象中定义了什么方法来创建一级目录？（D、E）
7. createNewFile B. exists C. createFolder
8. mkdirs E. mkdir

3、读写原始数据，一般采用什么流？（A、C）  
A InputStream B DataInputStream C OutputStream D BufferedInputStream

4、以下通过哪个类对象可以实现多个线程或进程中传递数据（ABCD）

A.PipedInputStream B.PipedOutputStrean C.PipedReader D. PipedWriter

5、以下哪些是定义在java.io包中的抽象类？（ A、C ）

A．InputStream B．PrintStream C．Reader D．FileInputStream

6、以下哪些是FileOutputSteram类的正确构造形式？（ A、B、D ）

A．FileOutputStream(FileDescriptor fd)

B．FileOutputStream(String n,boolean a)

C．FileOutputStream(boolean a)

D．FileOutputStream(File f)

7、下列哪些选项的内容可以作为 FileInputStream 类构造方法的参数？（A、B）

A. File B. String C. InputStream D. FileOutPutStream

8、 以下哪几个是字节流？（A、B）

A. InputStream B. OutputStream C. Read

D. Write

9、 读写原始数据，一般采用什么流?（A、C）

A.InputStream B.DataInputStream C.OutputStream D.BufferedInputStream

10、在java中，被看作为输出流的选项是（A、D）

A. OutputStream B. Reader C. InputStream D. Writer

11、以下关于File类的描述哪些是正确的？（A、C）

A.File类可以用于访问当前工作路径中的文件

B.当一个File类实例被构建时，对应的目录或文件在本地文件系统中被创建

C.File类可以用于访问本地文件系统中的目录或文件

D.当一个File类实例被垃圾回收器回收后，对应的目录或文件也被删除

1. 填空题（每空1分，共10分）
2. 当时用IO流操作文本文件时一般使用\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_对象。(FileReader、FileWriter)
3. 如果在对象序列化的时候不将某一个属性的数据保存在硬盘上面,采用\_\_\_\_\_\_\_\_\_\_关键字修饰该属性。(transient)
4. 能指定字符编码的I/O流类型是\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_。（InputStreamReader、 OutputStreamWriter）
5. 数据流根据编码格式上划分为 字节流 和 字符流 。
6. File类中实现创建文件夹功能的方法是 [mkdir](mk:@MSITStore:C:\Users\Administrator\Desktop\JDK1.6%20API帮助文档.CHM::/java/io/File.html#mkdir()) 。
7. JDK为每种类型的流提供了一些抽象类以供继承，请说出他们分别是哪些类\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_。(InputStream、OutputStream、Reader、Writer)
8. Java 系统的标准输入对象是 \_\_\_\_\_\_\_\_\_\_ ，标准输出对象有两个，分别是标准输出 \_\_\_\_\_\_\_\_\_\_ 和标准错误输出 \_\_\_\_\_\_\_\_\_\_ 。 (System.in、System.out、System.err)
9. FileInputStream 是\_\_\_\_\_\_\_流，BufferedWriter 是\_\_\_\_\_\_\_流，ObjectOutputStream 是\_\_\_\_\_\_\_流
10. Java中的IO流的四个基类：\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_。(InputStream 、OutputStream、Reader、Writer)
11. 在java.lang包中System有两个非常重要的类\_\_\_\_\_\_\_\_\_\_ 、\_\_\_\_\_\_\_\_\_分别是输入流和输出流类型的对象。(in、out)
12. 判断题
13. 当时用IO流操作文件之后不需要关闭流，因为jvm会协助关闭的。（ F ）
14. newline()方法是FileWiter类的方法。（F）
15. 处理流提供了缓冲功能，从而通过处理流输入输出数据效率更高。（正确）
16. 从内存往外部设备写数据，称为输出。把数据读到内存中，称为输入。（ T ）
17. 假设文件“a.txt”的长度为 100 字节，那么当正常运行语句“OutputStream os = new FileOutputStream(new File("a.txt"));”之后，文件“a.txt”的长度变为0字节。（ T ）
18. ByteArrayInutStream 和 ByteArrayOutputStream 对内存中的字节数组进行读写操作，属于字节流，属于处理流而不是节点流。 （ F ）
19. 实现 Serializable 接口的可以被序列化和反序列化。该接口中没有定义抽象方法，也没有定义常量。（ T ）
20. 序列化是指将字节序列转换成 Java 对象，只有实现了 Serializable 接口的类的对象才可以被序列化。（ F ）
21. BufferedInputStream 和 BufferedOutputStream 可以指定字符编码。（ F ）
22. Java IO 的各种流是非阻塞的，并且是面向缓冲的。（ F ）
23. 假设文件”a.txt”的长度为100字节，那么当正常运行语句”OutputStream f=new FileOutputStream(new File(“a.txt”));”之后，文件”a.txt”的长度变为0字节（ T ）
24. ByteArrayInutStream和ByteArrayOutputStream对内存中的字节数组进行读写操作，属于字节流，属于处理流而不是节点流。 （ F ）
25. 实现Serializable接口的可以被序列化和反序列化。该接口中没有定义抽象方法，也没有定义常量。（ T ）
26. 序列化是指将字节序列转换成Java对象，只有实现了Serializable接口的类的对象才可以被序列化。（ F ）
27. Java流的方向是先进先出的方式。（ T ）
28. Directory类提供了创建一个目录的方法。（ F）
29. RandomAccessFile类提供有删除磁盘文件的方法。（ F）
30. FileWriter类提供有操作基本数据类型的方法。（ F）

五、简答题

1、什么是java序列化，如何实现java序列化？（5分）

答案：

（1）序列化就是一种用来处理对象流的机制，所谓对象流也就是将对象的内容进行流化（将对象转换成二进制）。可以对流化后的对象进行读写操作，也可将流化后的对象传输于网络之间。（2）将需要序化的类实现Serializable接口就可以了，该接口没有任何方法，可以理解为一个标记，即表明这个类可以序列化。

2、节点流和处理流有什么区别？

答案：

节点流封装的是某些特定的数据源，如文件、字符串等。

字节流常用的类如FileInputStream、FileOutputStream、FileReader、FileWrite等

处理流封装的是其他流对象。

处理流常用的类如BufferOutputStream、BufferReader等

3、Java字节流和字符流的区别？

答案：Java 流在处理上分为字符流和字节流。字符流处理的单元为 2 个字节的 Unicode 字符，分别操作字符、字符数组或字符串，而字节流处理单元为 1 个字节，操作字节和字节数组。

Java 内用 Unicode 编码存储字符，字符流处理类负责将外部的其他编码的字符流和 java 内 Unicode 字符流之间的转换。而类 InputStreamReader 和 OutputStreamWriter 处理字符流和字节流的转换。字符流（一次可以处理一个缓冲区）一次操作比字节流（一次一个字节）效率高。

4、Java 中有几种类型的流？JDK 为每种类型的流提供了一些抽象类以供继承，请说出他们分别是哪些类？为了提高性能，可以采用什么流？

答案：

字节输入流：InputStream，字节输出流：OutputStream。

字符输入流：Reader，字符输出流：Writer

为了提高读写性能：

字节输入流：BufferedInputStream，字节输出流：BufferedOutputStream。

字符输入流：BufferedReader，字符输出流：BufferedWriter

5、 请简要说明PrintStream、BufferedWriter、PrintWriter各自的特点。

答案：

PrintStream类的输出功能非常强大，通常如果需要输出文本内容，都应该将输出流包装成 PrintStream 后进行输出。它还提供其他两项功能。与其他输出流不同，PrintStream 永远不会抛出 IOException ；而是，异常情况仅设置可通过 checkError 方法测试的内部标志。另外，为了自动刷新，可以创建一个 PrintStream。

BufferedWriter：将文本写入字符输出流，缓冲各个字符从而提供单个字符，数组和字符串的高效写入。通过 write() 方法可以将获取到的字符输出，然后通过 newLine() 进行换行操作。BufferedWriter 中的字符流必须通过调用 flush() 方法才能将其刷出去。并且 BufferedWriter() 只能对字符流进行操作。如果要对字节流操作，则使用 BufferedOutputStream 。

PrintWriter的 println() 方法自动添加换行，不会抛异常，若关心异常，需要调用 checkError() 方法看是否有异常发生，PrintWriter 构造方法可指定参数，实现自动刷新缓存(autoflush);

6、 什么是 Java 序列化，如何实现 Java 序列化？

答案：

Java对象的序列化指将一个java对象写入 IO 流中，与此对应的是，对象的反序列化则从 IO 流中恢复该 Java 对象。

如果要让某个对象支持序列化机制，则必须让它的类是可序列化的，为了让某个类是可序列化的，该类必须实现 Serializable 接口或 Externalizable 接口。

7、 试说明 Externalizable 和 Serializable 的区别。

答案：

1、Serializable序列化时不会调用默认的构造器，而Externalizable序列化时会调用默认构造器的！！！

2、Serializable：一个对象想要被序列化，那么它的类就要实现此接口，这个对象的所有属性（包括private属性、包括其引用的对象）都可以被序列化和反序列化来保存、传递。

Externalizable：它是Serializable接口的子类，有时我们不希望序列化那么多，可以使用这个接口，这个接口的 writeExternal() 和 readExternal() 方法可以指定序列化哪些属性。

8、什么是IO流？

答案：IO流用来处理设备之间的数据传输，Java对数据的操作是通过流的方式，Java用于操作流的类都在IO包中

9、 PrintStream、BufferedWriter、PrintWriter的比较?

答案：PrintStream类的输出功能非常强大，通常如果需要输出文本内容，都应该将输出流包装成PrintStream后进行输出。它还提供其他两项功能。与其他输出流不同，PrintStream 永远不会抛出 IOException;而是，异常情况仅设置可通过 checkError 方法测试的内部标志。另外，为了自动刷新，可以创建一个 PrintStream

　　BufferedWriter:将文本写入字符输出流，缓冲各个字符从而提供单个字符，数组和字符串的高效写入。通过write()方法可以将获取到的字符输出，然后通过newLine()进行换行操作。BufferedWriter中的字符流必须通过调用flush方法才能将其刷出去。并且BufferedWriter只能对字符流进行操作。如果要对字节流操作，则使用BufferedInputStream。

PrintWriter的println方法自动添加换行，不会抛异常，若关心异常，需要调用checkError方法看是否有异常发生，PrintWriter构造方法可指定参数，实现自动刷新缓存(autoflush);

10、 java中有几种类型的流?JDK为每种类型的流提供了一些抽象类以供继承，请说出他们分别是哪些类?

答案：字节输入流：InputStream,字节输出流：OutputStream

字符输入流：Reader，字符输出流：Writer

11、 字符流和字节流有什么区别？

答案：字符流以字符或者字符数组的形式读写数据,只能读写二进制文件;字节流能读写各种类型的数据

12、 什么是java序列化，如何实现java序列化?

答案：Java对象的序列化指将一个java对象写入OI流中，与此对应的是，对象的反序列化则从IO流中恢复该java对象。如果要让某个对象支持序列化机制，则必须让它的类是可序列化的，为了让某个类是可序列化的，该类必须实现Serializable接口或Externalizable接口

13、 FileInputStream和FileOutputStream是什么？

答案：这是在拷贝文件操作的时候，经常用到的两个类。在处理小文件的时候，它们性能表现还不错，在大文件的时候，最好使用BufferedInputStream (或 BufferedReader) 和 BufferedOutputStream (或 BufferedWriter)

14、什么叫对象序列化，什么是反序列化，实现对象序列化需要做哪些工作

答案：对象序列化，将对象以二进制的形式保存在硬盘上

反序列化；将二进制的文件转化为对象读取

实现serializable接口

15. Java中的所有的流可以分为几大类,它们的名字是什么,各代表什么

字节输入流 InputStream

字节输出流 OutputStream

字符输入流 Reader

字符输出流 Writer

所有流都是这四个流的子类

16.OutputStream里面的write()是什么意思,write(byte b[], int off, int len)这个方法里面的三个参数分别是什么意思

write将指定字节传入数据源

Byte b[ ]是byte数组

b[off]是传入的第一个字符

b[off+len-1]是传入的最后的一个字符

len是实际长度

17、什么是java序列化，如何实现java序列化？

答案：序列化的实现：将需要被序列化的类实现Serializable接口，该接口没有需要实现的方法，implements Serializable只是为了标注该对象是可被序列化的，然后使用一个输出流(如：FileOutputStream)来构造一个ObjectOutputStream(对象流)对象，接着，使用ObjectOutputStream对象的writeObject(Object obj)方法就可以将参数为obj的对象写出(即保存其状态)，要恢复的话则用输入流。

18、Java字节流和字符流的区别？

答案：Java 流在处理上分为字符流和字节流。字符流处理的单元为 2 个字节的 Unicode 字符，分别操作字符、字符数组或字符串，而字节流处理单元为 1 个字节，操作字节和字节数组。

Java 内用 Unicode 编码存储字符，字符流处理类负责将外部的其他编码的字符流和 java 内 Unicode 字符流之间的转换。而类 InputStreamReader 和 OutputStreamWriter 处理字符流和字节流的转换。字符流（一次可以处理一个缓冲区）一次操作比字节流（一次一个字节）效率高。

19、若通过ObjectOutputStream向一个文件中多次以追加方式写入object,为什么用ObjectInputStream读取这些object时会产生StreamCorruptedException?

答案：使用缺省的serializetion的实现时,一个ObjectOutputStream的构造和一个ObjectInputStream的构造必须一一对应.ObjectOutputStream的构造函数会向输出流中写入一个标识头,而ObjectInputStream会首先读入这个标识头.因此,多次以追加方式向一个文件中写入object时,该文件将会包含多个标识头.所以用ObjectInputStream来deserialize这个ObjectOutputStream时,将产生StreamCorruptedException.一种解决方法是可以构造一个ObjectOutputStream的子类,并覆盖writeStreamHeader()方法.被覆盖后的writeStreamHeader()方法应判断是否为首次向文件中写入object,是则调用super.writeStreamHeader();若否,即以追加方式写入object时,则应调用ObjectOutputStream.reset()方法.

六、程序实现题

1、编写一个函数将某一个文件的内容读取出来，另外编写一个函数将指定字符写入某一个文件中，其中文件是txt格式，文件的内容为字符。

（10分）

|  |
| --- |
| import java.io.File;  import java.io.FileNotFoundException;  import java.io.FileReader;  import java.io.FileWriter;  import java.io.IOException;  public class TestFileWriter {  public static void main(String[] args) throws Exception {  writeToFile();  readFromFile();  }    /\*\*  \* DOC 从文件里读取数据.  \*  \* @throws FileNotFoundException  \* @throws IOException  \*/  private static void readFromFile() throws FileNotFoundException, IOException {  File file = new File("E:\\helloworld.txt");// 指定要读取的文件  FileReader reader = new FileReader(file);// 获取该文件的输入流  char[] bb = new char[1024];// 用来保存每次读取到的字符  String str = "";// 用来将每次读取到的字符拼接，当然使用StringBuffer类更好  int n;// 每次读取到的字符长度  while ((n = reader.read(bb)) != -1) {  str += new String(bb, 0, n);  }  reader.close();// 关闭输入流，释放连接  System.out.println(str);  }    /\*\*  \* DOC 往文件里写入数据.  \*  \* @throws IOException  \*/  private static void writeToFile() throws IOException {  String writerContent = "hello world,你好世界";// 要写入的文本  File file = new File("E:\\helloworld.txt");// 要写入的文本文件  if (!file.exists()) {// 如果文件不存在，则创建该文件  file.createNewFile();  }  FileWriter writer = new FileWriter(file);// 获取该文件的输出流  writer.write(writerContent);// 写内容  writer.flush();// 清空缓冲区，立即将输出流里的内容写到文件里  writer.close();// 关闭输出流，施放资源  }  } |

2、 从磁盘读取一个文件到内存中，再打印到控制台

|  |
| --- |
| package com.xykj.lesson4;  import java.io.File;  import java.io.FileInputStream;  public class Tset4 {  public static void main(String[] args)  // 读取D:\notePad\aa.txt里面的内容  File file = new File("D:\\notePad\\aa.txt");  try {  // 创建读取流即输入流  FileInputStream fis = new FileInputStream(file);  int len = 0;  byte[] buf = new byte[1024];  StringBuffer sb = new StringBuffer();  // 把读取的数据添加到StringBuffer里面  while ((len = fis.read(buf)) != -1) {  sb.append(new String(buf, 0, len));  }  // 把StringBuffer里面的内容打印出来  System.out.println(sb);  fis.close();  } catch (Exception e) {  e.printStackTrace();  }  }  } |

3、编写程序，检测主机是否插入优盘，如有优盘插入，则打印提示信息。

int DiskNumber;

DiskNumber = File.listRoots().length;

System.out.println(DiskNumber);

while (true){

if (DiskNumber < File.listRoots().length){

System.err.println("有磁盘插入。");

break;

}

}

4、一个文件中保存了一行数据，数据用逗号分隔，请读取出数据，进行升序排序，然后将排好序的数据写入文件。

//从文件读取数据

FileReader reader = new FileReader("a.txt");

char[] array = new char[1024];

reader.read(array);

reader.close();

//将读到的数据转换为一个一个的数据

String str = new String(array);

String[] numbers;

numbers = str.split(",");

int[] number = new int[numbers.length];

for (int i=0; i<number.length; i++)

//加trim()为了去掉字符串两边的空字符。

number[i] = Integer.parseInt(numbers[i].trim());

//排序

for(int i=0; i<number.length-1; i++){

for (int j=0; j<number.length-i-1; j++){

if (number[j] > number[j+1]){

int t;

t = number[j];

number[j] = number[j+1];

number[j+1] = t;

}

}

}

//将排好序的数据写入到文件中

FileWriter writer = new FileWriter("a.txt", true); //不覆盖原有内容

for (int i=0; i<number.length; i++){

if (i != number.length-1)

writer.write(number[i]+",");

else

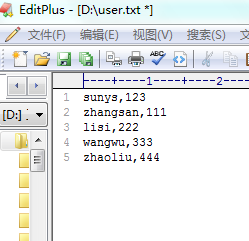
writer.write(number[i]+""); //最后一个数后面不加逗号。

}

writer.close();

5、模拟用户登录：

准备一个user.txt，里边添加一些测试数据



从控制台模拟输入用户名和密码，如果输入的用户名和密码在文本文档中存在则输出登录成功，如果不存在则输出用户名或密码错误。

（10分）

|  |
| --- |
| public static void main(String[] args) {      //让用户可以在控制台输入用户名和密码  Scanner input = new Scanner(System.in);  System.out.println("请输入用户名：");  String name = input.next();    //想要把hello.txt中的内容读取到内存中，并且展现在控制台上  BufferedReader br = null;  try {  FileReader fr = new FileReader("D:\\hello.txt");  br = new BufferedReader(fr);    String line = br.readLine();//readLine()读取一行    boolean flag = true;  while(line!=null){  //System.out.println(line);  //验证用户输入的用户名在此文件中是否存在  String[] sArray = line.split(",");  if(name.equals(sArray[0])){//证明用户名存在  System.out.println("请输入密码：");  String pwd = input.next();  if(pwd.equals(sArray[1])){  System.out.println("登录成功");  }else{  System.out.println("密码错误");  }  flag = false;  break;  }  line = br.readLine();  }  if(flag){  System.out.println("用户名不存在");  }    } catch (Exception e) {  e.printStackTrace();  } finally{  try {  br.close();  input.close();  } catch (IOException e) {  e.printStackTrace();  }  }    } |

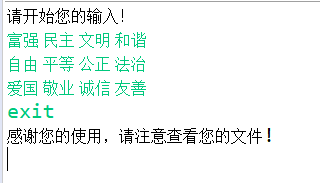
6、实现字符串和字节数组之间的相互转换。比如将字符串“Java IO程序示例”转换为字节数组，并将字节数组再转换回字符串。

|  |
| --- |
| **import** java.io.IOException;  **public** **class** TestConvert {  **public** **static** **void** main(String[] args) **throws** IOException {  // 准备一个字符串  String contents = "Java IO程序示例";  System.***out***.println(contents);  // String--->byte[]  **byte**[] buf = contents.getBytes();  // byte[]--->String  String contents2 = **new** String(buf, 0, buf.length);  System.***out***.println(contents2);  }  } |

7、请编写控制台程序，以行为单位接受输入的任意字符，输入【exit】结束输入(不区分大小写)，将输入的内容保存在任意的文本文件中即可，要求字符编码为【utf-8 no bom】，换行符为 【\n】。

|  |
| --- |
| **import** java.io.BufferedWriter;  **import** java.io.FileOutputStream;  **import** java.io.IOException;  **import** java.io.OutputStreamWriter;  **import** java.util.ArrayList;  **import** java.util.List;  **import** java.util.Scanner;  **public** **class** BasicReadWriteDemo {  **public** **static** **void** main(String[] args) **throws** IOException {  Scanner scanner = **new** Scanner(System.***in***);  List<String> contents = **new** ArrayList<>();  System.***out***.println("请开始您的输入！");  **while** (scanner.hasNextLine()) {  String line = scanner.nextLine();  **if** ("exit".equalsIgnoreCase(line)) {  **break**;  }  contents.add(line);  }  scanner.close();  System.***out***.println("感谢您的使用，请注意查看您的文件!");  *outputContents*(contents);  }  // 将接收到的输入内容保存到文件中  **private** **static** **void** outputContents(List<String> contents) **throws** IOException {  FileOutputStream fos = **null**;  OutputStreamWriter osw = **null**;  BufferedWriter bw = **null**;  **try** {  fos = **new** FileOutputStream("./test.txt");  osw = **new** OutputStreamWriter(fos, "utf-8");  bw = **new** BufferedWriter(osw);    **for** (String content : contents) {  bw.append(content);  bw.append('\n');  }  } **catch** (IOException e) {    e.printStackTrace();    } **finally** {  **if** (bw != **null**) {  bw.close();  }    **if** (osw != **null**) {  osw.close();  }    **if** (fos != **null**) {  fos.close();  }  }  }  } |

运行效果：



8、把一个D盘某一目录下的.txt文件复制到E盘某一目录下

（10分）

|  |
| --- |
| **public** **class** TestFileCopy {  **public** **static** **void** main(String[] args) **throws** IOException {  *copyDir*("D:/test1", "E:/test2");  }    //将srcDirName目录下的txt文件都拷贝到dstDirName目录下  **public** **static** **void** copyDir(String srcDirName,String dstDirName) **throws** IOException {  File file = **new** File(srcDirName);  File[] files = file.listFiles(**new** java.io.FileFilter() {  @Override  **public** **boolean** accept(File file) {  //判断文件的后缀是否为txt  **if**(file.getName().endsWith(".txt")) {  **return** **true**;  }  **return** **false**;  }  });    File dstDir = **new** File(dstDirName);  **if**(!dstDir.isDirectory()) {  dstDir.mkdirs();  }    **for** (File f : files) {  *copyFile*(f, dstDir);  }  }    //将文件拷贝到目标目录下  **public** **static** **void** copyFile(File srcFile,File dstDir) **throws** IOException {  String destFileName = dstDir.getAbsolutePath() + File.***separator*** + srcFile.getName();  **try**(FileInputStream fis = **new** FileInputStream(srcFile); FileOutputStream fos = **new** FileOutputStream(**new** File(destFileName));){  **byte**[] buffer = **new** **byte**[1024];  **int** c = 0;  **while** ((c = fis.read(buffer)) != -1) {  fos.write(buffer, 0, c);  }  }  }  } |

9、io流怎样读取文件的（10分）

|  |
| --- |
| File file = new File("d:/spring.txt"); try { Reader reader = new FileReader(file); BufferedReader buffered = new BufferedReader(reader); String data = null; while((data = buffered.readLine())!=null){ System.out.println(data); } } catch (FileNotFoundException e) { e.printStackTrace(); } catch (IOException e) { e.printStackTrace(); } |

10、实现为文本文件从C盘下data.txt的复制到D盘功能。

|  |
| --- |
| public class CopyFileDemo {  public static void main(String[] args) throws IOException {  File srcFile = new File("c:\\data.txt"); //源文件对象  File destFile = new File("d:\\data.txt"); //目标文件对象  if (!(destFile.exists())) { //判断目标文件是否存在  destFile.createNewFile(); //如果不存在则创建新文件  }    FileReader fr = new FileReader(srcFile);  BufferedReader br = new BufferedReader(fr);  FileWriter fw = new FileWriter(destFile,true);  BufferedWriter bw = new BufferedWriter(fw);    String msg=br.readLine();  while(msg!=null){  System.out.println(msg);  bw.write(msg);  bw.newLine();  msg=br.readLine();  }  System.out.println("复制成功！");  br.close();  fr.close();  bw.close();  fw.close();  }  } |