**IO流作业**

1. **填空题**
2. Java IO流可以分为 节点流 和处理流两大类，其中前者处于IO操作的第一线，所有操作必须通过他们进行。
3. 输入流的唯一目的是提供通往数据的通道，程序可以通过这个通道读取数据， read()

方法给程序提供了一个从输入流中读取数据的基本方法。

1. read方法从输入流中顺序读取源中的单个字节数据，该方法返回字节值(0-255之间的一个整数)，如果到达源的末尾，该方法返回 -1 。
2. Java系统的标准输入对象是System.in，标准输出对象有两个，分别是标准输出System.out和标准错误输出\_\_\_system.err\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。
3. Java IO体系中，\_\_\_\_\_\_inputstream\_\_\_\_\_\_\_\_\_\_\_\_是字节输入流，不仅提供了存取所有Java基础类型数据（如：int，double 等）和String的方法,也提供了提供存取对象的方法。
4. Java IO体系中，\_\_\_\_\_\_\_\_outputstream\_\_\_\_\_\_\_\_\_是字节输出流，提供了可以存取所有Java基础类型数据（如：int，double 等）和String的方法,但没有提供存取对象的方法。
5. \_\_\_\_\_objectinputstream\_\_\_\_\_\_是指将Java对象转换成字节序列，从而可以保存到磁盘上，也可以在网络上传输，使得不同的计算机可以共享对象。
6. **选择题**

|  |  |  |
| --- | --- | --- |
| **1.** | **使用Java IO流实现对文本文件的读写过程中，需要处理下列（ B ）异常。（选择一项）** | |
|  |  |  |
|  | **A** | ClassNotFoundException |
|  | **B.** | IOException |
|  | **C.** | SQLException |
|  | **D.** | RemoteException |

|  |  |  |
| --- | --- | --- |
| **2.** | **在Java的IO操作中，（ BD ）方法可以用来刷新流的缓冲。（选择两项）** | |
|  |  |  |
|  | **A** | void release() |
|  | **B.** | void close() |
|  | **C.** | void remove() |
|  | **D.** | void flush() |

|  |  |  |
| --- | --- | --- |
| **3.** | **在Java中，下列关于读写文件的描述错误的是（ B ）。（选择一项）** | |
|  |  |  |
|  | **A** | Reader类的read()方法用来从源中读取一个字符的数据 |
|  | **B.** | Reader类的read(int n )方法用来从源中读取一个字符的数据 |
|  | **C.** | Writer类的write(int n)方法用来向输出流写入单个字符 |
|  | **D.** | Writer类的write(String str)方法用来向输出流写入一个字符串 |

|  |  |  |
| --- | --- | --- |
| **4.** | **阅读下列文件定入的Java代码，共有（ C ）处错误。（选择一项）** | |
|  | **import** java.io.\*;  **public** **class** TestIO {  **public** **static** **void** main(String []args){  String str ="文件写入练习";  FileWriter fw = **null**; //1  **try**{  fw = **new** FileWriter("c:\mytext.txt"); //2  fw.writerToEnd(str); //3  }**catch**(IOException e){ //4  e.printStackTrace();  }**finally**{  //此处省略关闭流  }  }  } | |
|  |  |  |
|  | **A** | 0 |
|  | **B.** | 1 |
|  | **C.** | 2 |
|  | **D.** | 3 |

|  |  |  |
| --- | --- | --- |
| **5.** | **分析如下Java代码，有标注的四行代码中，有错误的是第（ D ）处。（选择一项）** | |
|  | **import** java.io.FileWriter;  **import** java.io.IOException;  **public** **class** Test {  **public** **static** **void** main(String[ ] args) {  String str = "Hello World";  FileWriter fw = **null**;  **try** {  fw = **new** FileWriter("c:\\hello.txt"); // 1  fw.write(str); // 2  } **catch** (IOException e) {  e.printStackTrace(); // 3  } **finally** {  fw.close(); // 4  }  }  } | |
|  |  |  |
|  | **A** | 1 |
|  | **B.** | 2 |
|  | **C.** | 3 |
|  | **D.** | 4 |

|  |  |  |
| --- | --- | --- |
| **6.** | **以下选项中关于如下代码的说法正确的是（ A D ）。（选择二项）** | |
|  | **public** **class** TestBuffered {  **public** **static** **void** main(String[] args) **throws** IOException {  BufferedReader br =  **new** BufferedReader(**new** FileReader("d:/bjsxt1.txt"));  BufferedWriter bw =  **new** BufferedWriter(**new** FileWriter("d:/bjsxt2.txt"));  String str = br.readLine();  **while**(str !=**null**){  bw.write(str);  bw.newLine();  str = br.readLine();  }  br.close();  bw.close();  }  } | |
|  |  |  |
|  | **A.** | 该类使用字符流实现了文件复制，将d:/bjsxt1.txt复制为d:/bjsxt2.txt |
|  | **B.** | FileReader和FileWriter是处理流，直接从文件读写数据 |
|  | **C.** | BufferedReader和BufferedWriter是节点流，提供缓冲区功能，提高读写效率 |
|  | **D.** | readLine()可以读取一行数据，返回值是字符串类型，简化了操作 |

|  |  |  |
| --- | --- | --- |
| **7.** | **InputStreamReader是转换流，可以将字节流转换成字符流，是字符流与字节流之间的桥梁。它的实现使用的设计模式是（ C ）。（选择一项）** | |
|  |  |  |
|  | **A.** | 工厂模式 |
|  | **B.** | 装饰模式 |
|  | **C.** | 适配器模式 |
|  | **D.** | 代理模式 |

1. **判断题**
2. 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。（ ）
3. 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。 （ ）
4. 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。（ ）
5. 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。（ ）
6. **简答题**
7. 输入流和输出流的联系和区别，字符流和字节流的联系和区别
8. 节点流和处理流的联系和区别
9. 列举常用的字节输入流和字节输出流并说明其特点，至少5对。
10. 说明缓冲流的优点和原理
11. 序列化的定义、实现和注意事项
12. **编码题**
13. 实现字符串和字节数组之间的相互转换。必如将字符串“山西优逸客”转换为字节数组，并将字节数组再转换回字符串。

**String** str ="山西优逸客";

/\*char[] ca = str.toCharArray();

for (char c : ca) {

System.out.print(c);

}\*/

**byte**[] b = str.getBytes();

**String** string = **new** String(b);

**System**.***out***.println(string.toString());

1. 实现字节数组和任何基本类型和引用类型执行的相互转换

提示：使用ByteArrayInutStream和ByteArrayOutputStream。

1. 分别使用文件流和缓冲流复制一个长度大于100MB的视频文件，并观察效率的差异。
2. 复制文件夹d:/sxtjava下面所有文件和子文件夹内容到d:/sxtjava2。

提示：涉及单个文件复制、目录的创建、递归的使用

1. **可选题**
2. 使用IO包中的类读取D盘上exam.txt文本文件的内容，每次读取一行内容，将每行作为一个输入放入ArrayList的泛型集合中并将集合中的内容使用加强for进行输出显示。
3. 假设从入学开始所有书写的Java类代码都在d:/sxtjava文件夹下，包括多级子文件夹。使用IO流获取从入学开始，到目前为止已经写了多少行Java代码。

提示：其实就是获取d:/sxtjava文件夹及其子文件夹下的所有.java文件，使用readLine()读取其中每一行，每读取一行，行数加1。所有的文件读取完毕，得到总共已经写的Java代码行数。需要结合递归实现。

1. 由控制台按照固定格式输入学生信息，包括学号，姓名，年龄信息，当输入的内容为exit退出；将输入的学生信息分别封装到一个Student对象中，再将每个Student对象加入到一个集合中，要求集合中的元素按照年龄大小正序排序；最后遍历集合，将集合中学生信息写入到记事本，每个学生数据占单独一行。