重庆科技学院

上机实验报告(程序设计类)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | 面向对象的序设计方法 | | 实验题目 | **输入输出处理** | | |
| 机房编号 | I301 | | 上机时间 | 2018-11-12 | | |
| 指导老师 | 王双明 | | 报告序号 | 4 | 上机成绩 | . |
| 学生姓名 | 袁阊越 | 学号 | 2018520612 | 教学班级 | 计科升2016 | |
| **一、上机目的和要求:**  1. 了解File类及常用方法；  2. 掌握字节流的使用方法；  3. 掌握字符流的使用方法；  4. 掌握对象序列化方法。 | | | | | | |
| **二、程序开发环境:**  处理器：Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz (8 CPUs), ~3.4GHz  操作系统：Windows 7 专业版 32-bit  内存：4GB  JDK：jdk1.6.0\_04  软件：eclipse | | | | | | |
| **三、上机内容(老师布置的具体任务):**  1. 使用Java语言提供的的输入、输出处理功能将一个文本文件的内容按行读出，每读出一行就顺序添加行号，并写入到另一个文件中。  2. 编程实现：使用带缓冲区的字节流拷贝一个文本文件。  3. 编程实现：使用带缓冲区的字符流拷贝一个文本文件。  **4. 利用Java输入/输出功能编程实现一个简易通讯录管理程序。每条信息至少包含：姓名、性别、手机号。程序能实现添加、修改、查找、删除等功能。**  5. 采用面向对象的设计思想，设计FileHelper类。FileHelper类中实现3个功能，依次是文件的打开操作、文件的保存操作、文件的拷贝操作，并进行测试。完善下列程序。  import java.io.\*;  public class FileHelper {  public static String openFile(String fileName){ //打开文件  StringBuffer sb=null;  FileInputStream fis=null;  try {  fis= \_\_\_\_\_\_\_\_\_\_\_\_\_ ; //实例化输入流对象  byte b[]=new byte[1024];  int len;  sb=new StringBuffer();  while( ( )!=-1 ){ //读文件并判断是否到达文件尾  String str=new String(b,0,len);  sb.append(str);  }    } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }finally{  try {  //如输入流非空则关闭输入流  } catch (IOException e) {  e.printStackTrace();  }  }  return sb.toString();  }  public static boolean saveFile(String fileName,String content){  boolean state=false;  FileOutputStream fos=null;  try {  fos=\_\_\_\_\_\_\_\_\_\_\_\_; //实例化输出流对象  //把content写入到文件中  state=true;  } catch (Exception e) {    e.printStackTrace();  }finally {  try {  //如输出流非空则关闭输出流  } catch (IOException e) {  e.printStackTrace();  }  }  return state;  }  public static boolean copyFile(String sourceFileName,String destinationFifleName){  boolean sate =false;  InputStream fis=null;  OutputStream fos=null;    try {  fis=new FileInputStream(sourceFileName);  fos=new FileOutputStream(destinationFifleName);  int len;  byte buffer[]=new byte[1024];  //此处请填写多行  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  sate =true;    } catch (Exception e) {  // TODO Auto-generated catch block  e.printStackTrace();  }finally{  try {  if(fis!=null) fis.close();  if(fos!=null) fos.close();  } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }  return sate;  }  }  6. 请把第5题的FileHelper.java文件复制到当前工程文件夹中，阅读下列程序，写出运行结果，并思考：  （1）变量length的单位是什么，如何将其转换为KB；  （2）如何将变量lastModifiedTime的结果转换为**yyyy/MM/dd HH：mm：SS**格式。  （3）思考变量path和absolutePath的区别是什么。  import java.io.File;  public class TestFile {  public static void main(String[] args) {  File file=new File("src\\FileHelper.java");    System.out.println(file.exists());    String name=file.getName();  String parent=file.getParent();  String absolutePath=file.getAbsolutePath();  String path=file.getPath();  long length =file.length();  long lastModifiedTime=file.lastModified();    System.out.println("Name:"+name);  System.out.println("Parent:"+parent);  System.out.println("AbsolutePath:"+absolutePath);  System.out.println("Path:"+path);  System.out.println("Length:"+length);  System.out.println("LastModifiedTime:"+lastModifiedTime);    }  }  7.（**选做**）利用面向对象的设计思想，创建学生类，学生信息包括：学号、姓名、出生日期。编写程序实现下列功能：  （1） 按表1所示格式，输入学生信息并写入到文件中，要求学生信息至少包含：学号、姓名、出生日期。  表1 学生信息   |  |  |  | | --- | --- | --- | | 学号 | 姓名 | 出生日期 | | 201644001 | 张三 | 1997年2月3日 | | 201644002 | 李四 | 1998年11月11日 | | 201644003 | 王五 | 1996年3月2日 | | 201644004 | 赵六 | 1996年12月12日 | | 201644005 | 周正 | 1997年10月10日 |   （2） 写程序读出题（1）所得的文件，把读到的学生信息保存到List集合（ArrayList或者LinkedList）中，并输出List集合的内容到屏幕（格式参考表1）。  8. （**选做**）使用RandomAccessFile类将一个文本文件中的内容逆序输出。 | | | | | | |
| **四、上机调试中出现的错误信息、错误原因及解决办法:**  报错：java.io.IOException: Negative seek offset  原因：RandomAccessFile调用seek方法的时候出现偏移  解决：正确调用seek（）方法  报错： java.io.IOException: Stream closed  原因：IO流已经关闭过  解决：删除多余的close（）； | | | | | | |
| **五、上机实验中的收获及心得:**  在做Java类和对象的设计的实验前，我以为不会难做，就像以前做物理实验一样，做完实验，然后两下子就将实验报告做完。直到做完测试实验时，我才明白其实并不容易做，但学到的知识与难度成正比，使我受益匪浅。  　　在做实验的过程中我还学会了面向对象的编程思想、类与对象。类的封装性、继承性和多态性的作用。成员变量和成员方法的特性、构造方法、toString方法、equals方法的使用。this、super、final、static等关键字的使用。接口及接口对象的使用。四种访问控制权限private、default、protected、public。  　　透过这次Java类和对象的设计的实验，使我学到了不少实用的知识，更重要的是，做实验的过程，思考问题的方法，这与做其他的实验是通用的，真正使我们受益匪浅。 | | | | | | |

建议：正文采用五宋体，代码用小五号Times new Roman字体

附录：

**一、程序流程图**

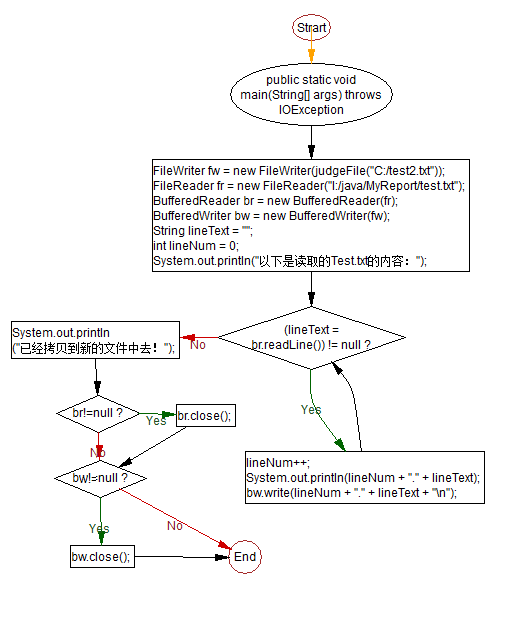


图1 第一题

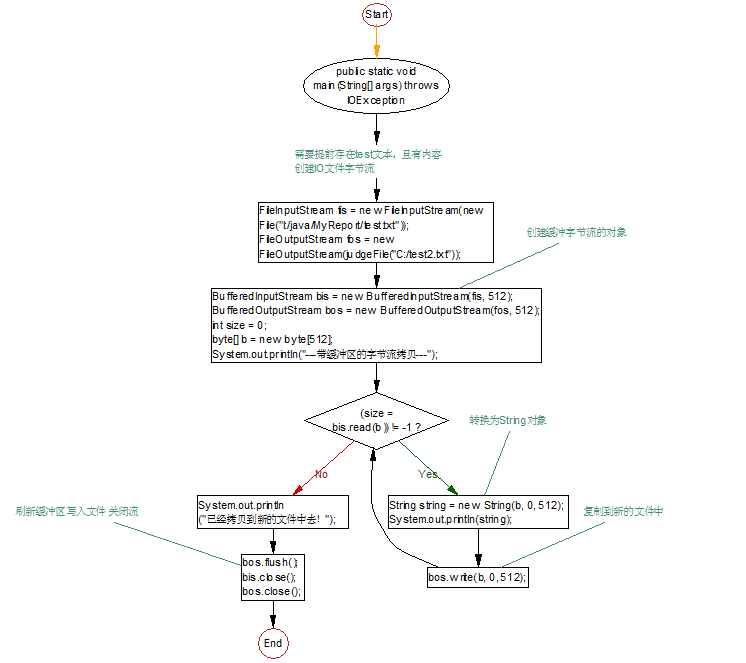


图2 第二题

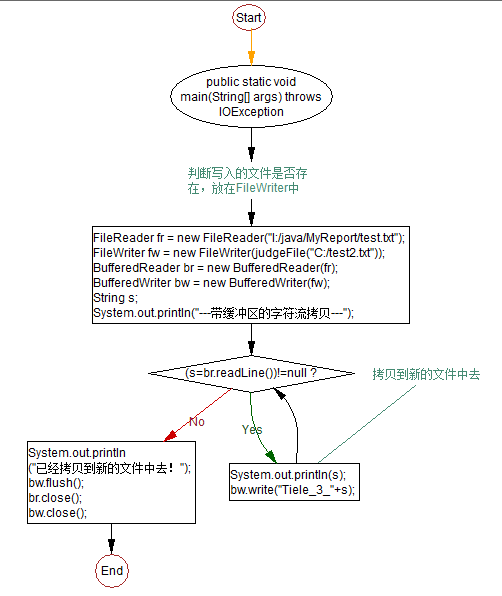


图3 第三题

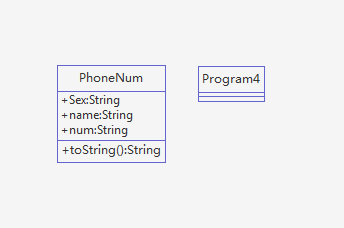


图4 第四题

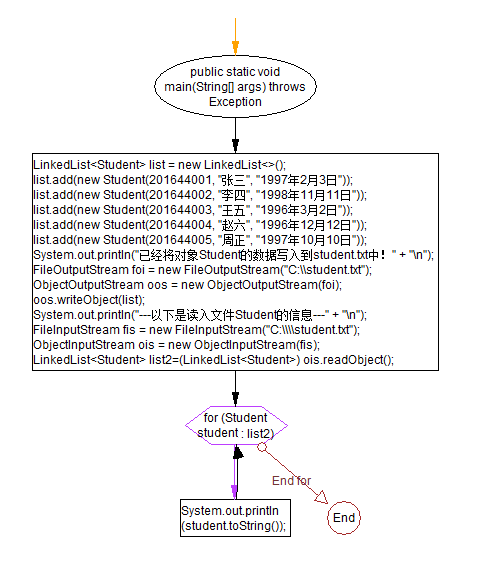


图5 第七题

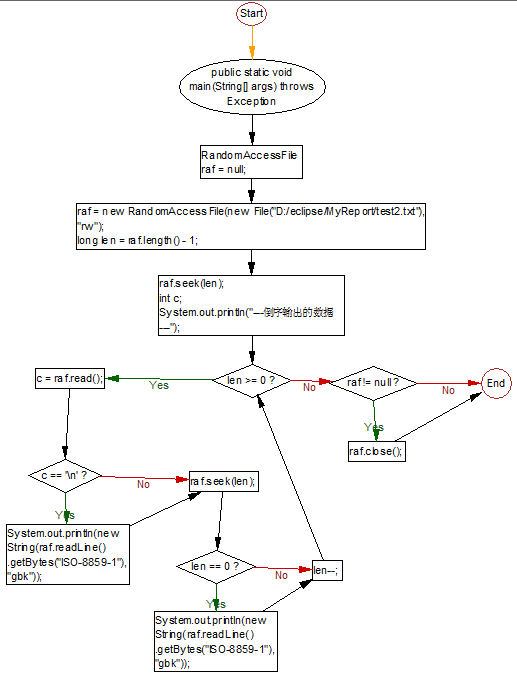


图6 第八题

**二、主要数据结构和变量说明**

第一题：

String lineText = ""; 用于接受读取的数据

**int** lineNum = 0; 统计行号

第二题：

**int** size = 0; 用于接收读取的数据

**byte**[] b = **new** **byte**[512]; 用于暂时存放数据的中转站

String string = **new** String(b, 0, 512); 将int数据转换为字符串类型

第三题：

String s; 用于接收读取的数据

第四题：

**public** String name; 姓名

**public** String Sex; 性别

**public** String num; 手机号码

LinkedList<PhoneNum> list 用于存放PhoneNum的java对象

LinkedList<PhoneNum> list2 用于接收封装后方法调用接收的list

第五题：

content.getBytes() 装换string类型到byte方便写入

**int** len = 0 用于接收读取的数据

**byte** buffer[] = **new** **byte**[1024];用于暂时存放数据的中转站

第六题：

(**int**)((length/1024)+1) b转换kb

第七题：

LinkedList<Student> list 用于存放Student的java对象

第八题：

**long** len = raf.length() - 1; 获取将文件的最后位置

**int** c ; 用于接收读取的数据

1. **上机调试后的源程序及注释**

**第一题**

**package** pers.ycy.report.test4;

**import** java.io.\*;

**public** **class** Program1 {

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

// 判断写入的文件是否存在，放在FileWriter中

// 需要提前存在test文本，且有内容

//创建IO文件字符流

FileWriter fw = **new** FileWriter(judgeFile("C:/test2.txt"));

FileReader fr = **new** FileReader("I:/java/MyReport/test.txt");

// 创建缓冲字符流的对象

BufferedReader br = **new** BufferedReader(fr);

BufferedWriter bw = **new** BufferedWriter(fw);

String lineText = "";

**int** lineNum = 0;

System.out.println("以下是读取的Test.txt的内容：");

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

lineNum++;

System.out.println(lineNum + "." + lineText);

// 利用BufferedWriter写入新的文件中

bw.write(lineNum + "." + lineText + "\n");

}

System.out.println("已经拷贝到新的文件中去！");

// 关闭IO流

**if** (br!=**null**) {

br.close();

}

**if** (bw!=**null**) {

bw.close();

}

}

**public** **static** File judgeFile(String s) **throws** IOException {

File file = **new** File(s);

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

file.createNewFile();

**return** file;

}

}

**第二题**

**package** pers.ycy.report.test4;

**import** java.io.\*;

**public** **class** Program2 {

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

// 需要提前存在test文本，且有内容

//创建IO文件字节流

FileInputStream fis = **new** FileInputStream(**new** File("I:/java/MyReport/test.txt"));

FileOutputStream fos = **new** FileOutputStream(judgeFile("C:/test2.txt"));

// 创建缓冲字节流的对象

BufferedInputStream bis = **new** BufferedInputStream(fis, 512);

BufferedOutputStream bos = **new** BufferedOutputStream(fos, 512);

**int** size = 0;

**byte**[] b = **new** **byte**[512];

System.out.println("---带缓冲区的字节流拷贝---");

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

//转换为String对象

String string = **new** String(b, 0, 512);

System.out.println(string);

//复制到新的文件中

bos.write(b, 0, 512);

}

System.out.println("已经拷贝到新的文件中去！");

//刷新缓冲区写入文件 关闭流

bos.flush();

bis.close();

bos.close();

}

**public** **static** File judgeFile(String s) **throws** IOException {

File file = **new** File(s);

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

file.createNewFile();

**return** file;

}

}

**第三题**

**package** pers.ycy.report.test4;

**import** java.io.\*;

**public** **class** Program3 {

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

// 判断写入的文件是否存在，放在FileWriter中

FileReader fr = **new** FileReader("I:/java/MyReport/test.txt");

FileWriter fw = **new** FileWriter(judgeFile("C:/test2.txt"));

BufferedReader br = **new** BufferedReader(fr);

BufferedWriter bw = **new** BufferedWriter(fw);

String s;

System.out.println("---带缓冲区的字符流拷贝---");

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

//拷贝到新的文件中去

System.out.println(s);

bw.write("Tiele\_3\_"+s);

}

System.out.println("已经拷贝到新的文件中去！");

bw.flush();

br.close();

bw.close();

}

**public** **static** File judgeFile(String s) **throws** IOException {

File file = **new** File(s);

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

file.createNewFile();

**return** file;

}

}

**第四题**

**package** pers.ycy.report.test4;

**import** java.io.Serializable;

**public** **class** PhoneNum **implements** Serializable {

**public** String name;

**public** String Sex;

**public** String num;

**public** PhoneNum(String name, String sex, String num) {

**super**();

**this**.name = name;

Sex = sex;

**this**.num = num;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getSex() {

**return** Sex;

}

**public** **void** setSex(String sex) {

Sex = sex;

}

**public** String getNum() {

**return** num;

}

**public** **void** setNum(String num) {

**this**.num = num;

}

@Override

**public** String toString() {

// return "姓名：" + name + "\n" + "性别：" + Sex + "\n" + "电话号码：" + num + "\n";

**return** "姓名：" + name + " " + "性别：" + Sex + " " + "电话号码：" + num + " ";

}

}

**package** pers.ycy.report.test4;

**import** java.io.\*;

**import** java.util.\*;

**public** **class** Program4 {

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

LinkedList<PhoneNum> list = **new** LinkedList<>();

// 添加到通讯录java对象的集合中去

addNum(list, "李四", "男", "18166334763");

addNum(list, "张三", "女", "15199827265");

addNum(list, "王五", "男", "15199827265");

// 将对象内容写入到文件中去

writeToFile(list);

// 读取原始通讯录数据

System.out.println("---以下是添加到通讯录的数据---" + "\n");

readAddress(list);

// 根据姓名修改通讯录信息

System.out.println("\n" + "---以下是修改后的通讯录数据---" + "\n");

upDataAddress(list, "张三", "123");

// 根据姓名删除通讯录信息

System.out.println("\n" + "---以下是删除后的通讯录数据---" + "\n");

deleteAddress(list, "张三");

// 根据姓名查询通讯录信息

System.out.println("\n" + "---以下是查询的通讯录数据---" + "\n");

queryAddress(list, "李四");

}

**public** **static** PhoneNum addNum(LinkedList<PhoneNum> list, String name, String sex, String number) {

PhoneNum num = **new** PhoneNum(name, sex, number);

list.add(num);

**return** num;

}

**public** **static** **void** readAddress(LinkedList<PhoneNum> list) {

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

System.out.println(list.get(i).toString());

}

}

**public** **static** **void** writeToFile(LinkedList<PhoneNum> list) **throws** Exception {

FileOutputStream fos = **new** FileOutputStream("C:\\test2.txt");

ObjectOutputStream oos = **new** ObjectOutputStream(fos);

// 创建输出流对象 可以实现将对象的内容写到文件中去

oos.writeObject(list);

}

**public** **static** **void** upDataAddress(LinkedList<PhoneNum> list, String name, String num) **throws** Exception {

FileInputStream fis = **new** FileInputStream("C:\\test2.txt");

// 创建输入流对象 了可以实现读取文件内容

ObjectInputStream ois = **new** ObjectInputStream(fis);

LinkedList<PhoneNum> list2 = (LinkedList<PhoneNum>) ois.readObject();

**for** (**int** i = 0; i < list2.size(); i++) {

**if** (list2.get(i).getName().equals(name)) {

list2.get(i).setNum(num);

System.out.println(list2.get(i).getName() + "的号码已经更新为" + num + "!" + "\n");

}

}

readAddress(list2);

// 变化后的数据重新存到文件中

writeToFile(list2);

}

**public** **static** **void** deleteAddress(LinkedList<PhoneNum> list, String name) **throws** Exception {

FileInputStream fis = **new** FileInputStream("C:\\test2.txt");

// 创建输入流对象 了可以实现读取文件内容

ObjectInputStream ois = **new** ObjectInputStream(fis);

LinkedList<PhoneNum> list2 = (LinkedList<PhoneNum>) ois.readObject();

**for** (**int** i = 0; i < list2.size(); i++) {

**if** (list2.get(i).getName().equals(name)) {

System.out.println(list2.get(i).getName() + "的号码已经删除!" + "\n");

list2.remove(list2.get(i));

}

}

readAddress(list2);

// 变化后的数据重新存到文件中

writeToFile(list2);

}

**public** **static** **void** queryAddress(LinkedList<PhoneNum> list, String name) **throws** Exception {

FileInputStream fis = **new** FileInputStream("C:\\test2.txt");

// 创建输入流对象 了可以实现读取文件内容

ObjectInputStream ois = **new** ObjectInputStream(fis);

LinkedList<PhoneNum> list2 = (LinkedList<PhoneNum>) ois.readObject();

**for** (PhoneNum phoneNum : list2) {

**if** (phoneNum.getName().equals(name)) {

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

}

}

// 变化后的数据重新存到文件中

writeToFile(list2);

}

}

**第五题**

**package** pers.ycy.report.test4;

**import** java.io.\*;

**public** **class** Program5 {

**public** **static** String openFile(String fileName) { // 打开文件

StringBuffer sb = **null**;

FileInputStream fis = **null**;

**try** {

fis = **new** FileInputStream(fileName); // 实例化输入流对象

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

**int** len;

sb = **new** StringBuffer();

**while** ((len = fis.read(b)) != -1) { // 读文件并判断是否到达文件尾

String str = **new** String(b, 0, len);

sb.append(str);

System.out.println(sb);

}

} **catch** (Exception e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **finally** {

**try** {

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

fis.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**return** sb.toString();

}

**public** **static** **boolean** saveFile(String fileName, String content) {

**boolean** state = **false**;

FileOutputStream fos = **null**;

**try** {

fos = **new** FileOutputStream(fileName); // 实例化输出流对象

// 把content写入到文件中

fos.write(content.getBytes());

state = **true**;

} **catch** (Exception e) {

e.printStackTrace();

} **finally** {

**try** {

// 如输出流非空则关闭输出流

**if** (fos != **null**)

fos.close();

} **catch** (Exception e2) {

// **TODO**: handle exception

}

}

**return** state;

}

**public** **static** **boolean** copyFile(String sourceFileName, String destinationFifleName) {

**boolean** sate = **false**;

InputStream fis = **null**;

OutputStream fos = **null**;

**try** {

fis = **new** FileInputStream(sourceFileName);

fos = **new** FileOutputStream(destinationFifleName);

**int** len = 0;

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

// 此处请填写多行

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

fos.write(buffer, 0, len);

}

sate = **true**;

} **catch** (Exception e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

} **finally** {

**try** {

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

fis.close();

**if** (fos != **null**)

fos.close();

} **catch** (IOException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

}

**return** sate;

}

}

**第六题**

**package** pers.ycy.report.test4;

**import** java.io.\*;

**import** java.text.SimpleDateFormat;

**public** **class** TestFile {

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

File file=**new** File("D:\\eclipse\\MyReport\\src\\pers\\ycy\\report\\test4\\Program5.java");

System.out.println(file.exists());

String name=file.getName();

String parent=file.getParent();

String absolutePath=file.getAbsolutePath();

String path=file.getPath();

**long** length =file.length();

**long** lastModifiedTime=file.lastModified();

System.out.println("Name:"+name);

System.out.println("Name:"+name);

System.out.println("Parent:"+parent);

System.out.println("AbsolutePath:"+absolutePath);

System.out.println("Path:"+path);

System.out.println("Length:"+length);

System.out.println("Length:"+(**int**)((length/1024)+1)+"kb");

System.out.println("LastModifiedTime:"+lastModifiedTime);

SimpleDateFormat dateFormat = **new** SimpleDateFormat("yyyy/MM/dd HH：mm：SS");

System.out.println("LastModifiedTime:"+dateFormat.format(lastModifiedTime));

}

}

（1）变量length的单位是什么，如何将其转换为KB；

**单位为B (文件的大小/1024)+1**

1. 如何将变量lastModifiedTime的结果转换为**yyyy/MM/dd HH：mm：SS**格式。

**使用SimpleDataFormat 转换**

**SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy/MM/dd HH：mm：SS");**

1. 思考变量path和absolutePath的区别是什么。

**Path是相对位置**

**absolutePath 是绝对位置**

**第七题**

**package** pers.ycy.report.test4;

**import** java.io.Serializable;

**public** **class** Student **implements** Serializable {

**private** **int** Id;

**private** String name;

**private** String born;

**public** Student(**int** id, String name, String born) {

**super**();

Id = id;

**this**.name = name;

**this**.born = born;

}

**public** **int** getId() {

**return** Id;

}

**public** **void** setId(**int** id) {

Id = id;

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** String getBorn() {

**return** born;

}

**public** **void** setBorn(String born) {

**this**.born = born;

}

@Override

**public** String toString() {

**return** Id + " " + name + " " + born;

}

}

**package** pers.ycy.report.test4;

**import** java.io.\*;

**import** java.util.LinkedList;

**public** **class** Program7 {

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

LinkedList<Student> list = **new** LinkedList<>();

list.add(**new** Student(201644001, "张三", "1997年2月3日"));

list.add(**new** Student(201644002, "李四", "1998年11月11日"));

list.add(**new** Student(201644003, "王五", "1996年3月2日"));

list.add(**new** Student(201644004, "赵六", "1996年12月12日"));

list.add(**new** Student(201644005, "周正", "1997年10月10日"));

System.out.println("已经将对象Student的数据写入到student.txt中！" + "\n");

FileOutputStream foi = **new** FileOutputStream("C:\\student.txt");

ObjectOutputStream oos = **new** ObjectOutputStream(foi);

oos.writeObject(list);

System.out.println("---以下是读入文件Student的信息---" + "\n");

FileInputStream fis = **new** FileInputStream("C:\\\\student.txt");

ObjectInputStream ois = **new** ObjectInputStream(fis);

LinkedList<Student> list2=(LinkedList<Student>) ois.readObject();

**for** (Student student : list2) {

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

}

}

}

第八题

**package** pers.ycy.report.test4;

**import** java.io.\*;

**import** java.io.RandomAccessFile;

**public** **class** Program8 {

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

RandomAccessFile raf = **null**;

**try** {

//拿到文件 设置可读可写

raf = **new** RandomAccessFile(**new** File("D:/eclipse/MyReport/test2.txt"), "rw");

**long** len = raf.length() - 1;

//将指针放到最后

raf.seek(len);

**int** c ;

System.out.println("---倒序输出的数据---");

**while** (len >= 0) {

//将文件读取到的数据给c

c = raf.read();

//通过\n读每行的数据输出并

**if** (c == '\n') {

System.out.println(**new** String(raf.readLine().getBytes("ISO-8859-1"), "gbk"));

}

//将指针放到下一行的最后位置

raf.seek(len);

//当指针读到0（第一行开始位置）输出第一行

**if** (len == 0) {

System.out.println(**new** String(raf.readLine().getBytes("ISO-8859-1"), "gbk"));

}

len--;

}

} **catch** (IOException e) {

e.printStackTrace();

} **finally** {

**try** {

**if** (raf != **null**)

raf.close();

} **catch** (IOException e) {

e.printStackTrace();

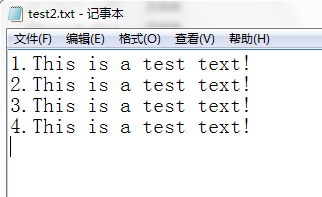
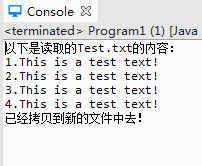
}

}

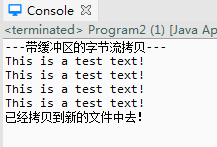
}

}

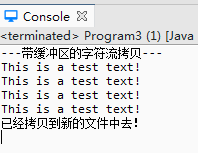
**四、程序运行结果截图及还存在的问题**



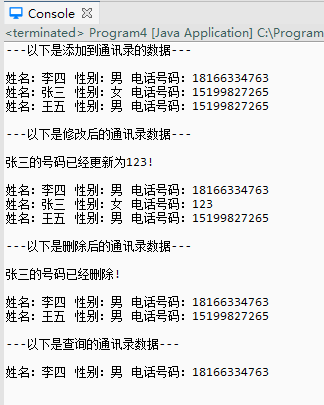
运行结果 第1题



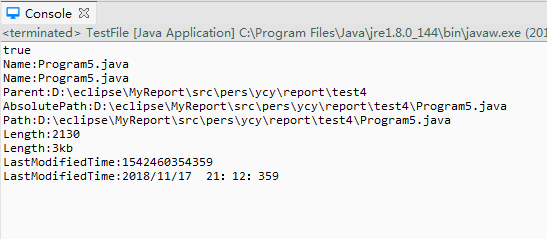
运行结果 第2题



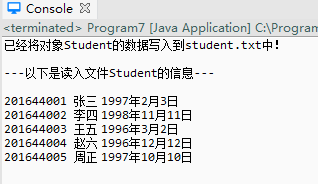
运行结果 第3题



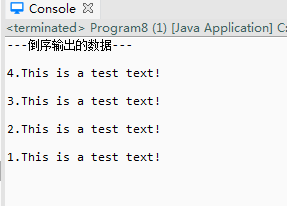
运行结果 第4题



运行结果 第6题



运行结果 第7题



运行结果 第8题