实验四 **Java**中的输入机制

实验日期： 2017 年 4 月 7 日 班级： 软件1501

学号（后四位）：\_\_\_\_\_0314\_\_\_\_\_ 姓名：\_\_\_马亚卿\_\_\_\_\_\_\_\_\_\_ 成绩：

1. 实验目的

1 Java如何操作文件。

2 了解Java中的输入机制：如何从控制台输入，如何用InputStream和Reader显示文件中的内容。

1. 实验内容

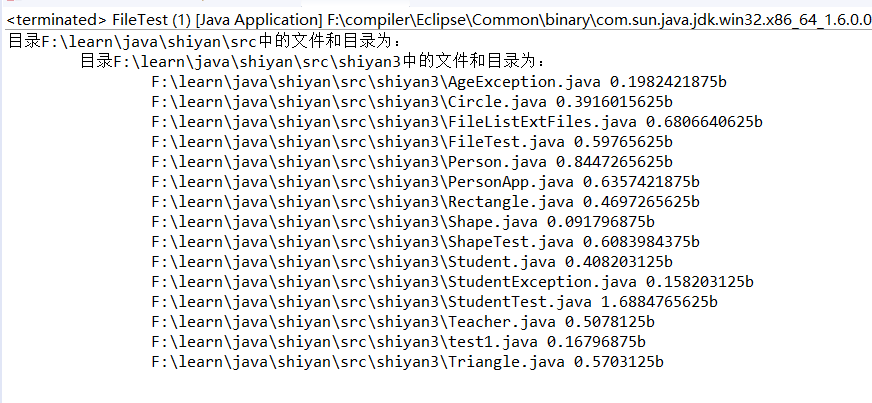
编写能够满足如下条件的程序：

1 递归遍历目录，显示其中的文件名和目录名。若为文件，则直接显示文件的大小；若为目录，则显示目录中的文件和子目录，并显示目录所含的所有文件的大小。注意显示文件大小的单位（KB或MB）。

代码：

|  |
| --- |
| import java.io.File;  public class FileTest {  void printFile(File f,int j){  for(int i =0;i<j;i++){  System.out.print("\t");  }  if(f.isDirectory()){  System.out.println("目录" + f + "中的文件和目录为：");    try {  for(File i : f.listFiles()){  printFile(i,j+1);  }  } catch (Exception e) {  }    }  else{  System.out.println(f + " "+ f.length()/1024.0 + "kb");  }  }  public static void main(String[] args) {  FileTest a = new FileTest();  File f = new File("C:\\Users\\MYQ\\Desktop\\test\\sdf\\src\\sdf");  a.printFile(f,0);  }  } |

运行结果：

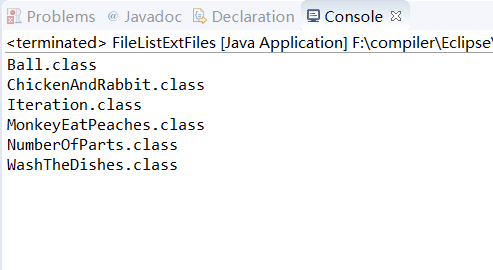


2.列出指定目录下文件名包含某一字符串的文件。

代码：

|  |
| --- |
| import java.io.File;  import java.io.FilenameFilter;  public class FileListExtFiles {  public static void list(File aFile,final String ext){  if(!aFile.isDirectory()) return;  String [] files = aFile.list(new FilenameFilter(){  @Override  public boolean accept(File dir, String name) {  // TODO Auto-generated method stub  if(new File(dir,name).isDirectory()) return false;  return name.endsWith(ext);  }    });  if(files == null) return;  for(String i : files)  System.out.println(i);  }  public static void main(String[] args) {  // TODO Auto-generated method stub  list(new File("F:\\learn\\java\\课后题"),"class");  }  } |

运行结果：

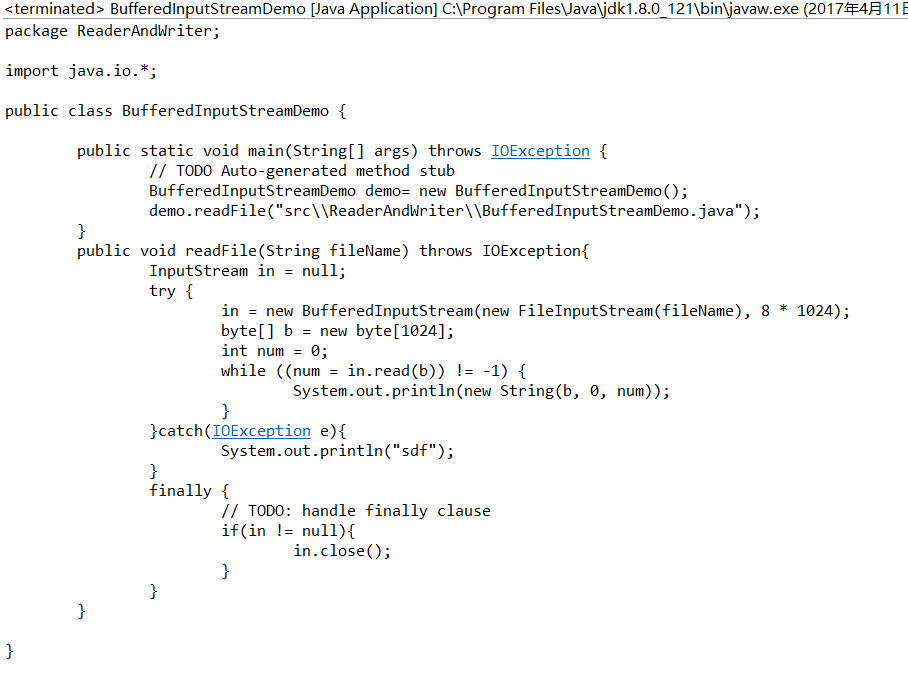


3 用InputStream的子类读入一个英文文本文件，并用System.out显示其中的内容。

代码：

|  |
| --- |
| public class BufferedInputStreamDemo {  public static void main(String[] args) throws IOException {  // TODO Auto-generated method stub  BufferedInputStreamDemo demo= new BufferedInputStreamDemo();  demo.readFile("src\\ReaderAndWriter\\BufferedInputStreamDemo.java");  }  public void readFile(String fileName) throws IOException{  InputStream in = null;  try {  in = new BufferedInputStream(new FileInputStream(fileName), 8 \* 1024);  byte[] b = new byte[1024];  int num = 0;  while ((num = in.read(b)) != -1) {  System.out.println(new String(b, 0, num));  }  }catch(IOException e){  System.out.println("sdf");  }  finally {  // TODO: handle finally clause  if(in != null){  in.close();  }  }  }  } |

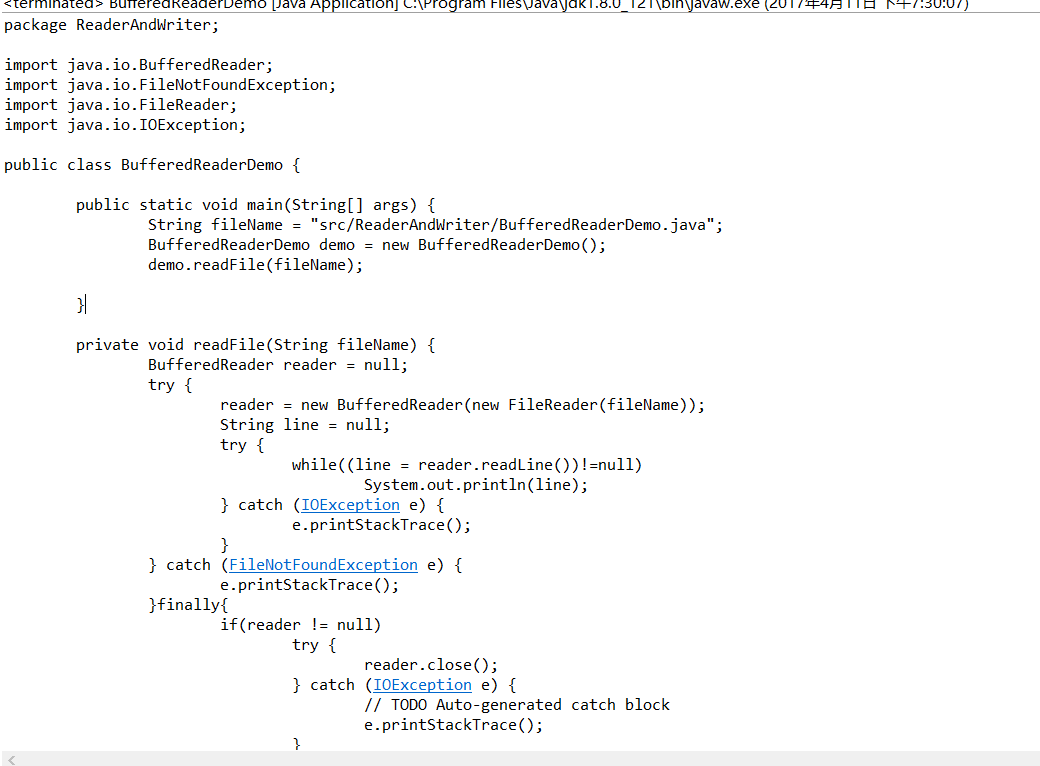
运行结果：



4 用Reader的子类读入一个一个字符文件，并用System.out显示其中的内容。

|  |
| --- |
| package ReaderAndWriter;  import java.io.BufferedReader;  import java.io.FileNotFoundException;  import java.io.FileReader;  import java.io.IOException;  public class BufferedReaderDemo {  public static void main(String[] args) {  String fileName = "src/ReaderAndWriter/BufferedReaderDemo.java";  BufferedReaderDemo demo = new BufferedReaderDemo();  demo.readFile(fileName);  }  private void readFile(String fileName) {  BufferedReader reader = null;  try {  reader = new BufferedReader(new FileReader(fileName));  String line = null;  try {  while((line = reader.readLine())!=null)  System.out.println(line);  } catch (IOException e) {  e.printStackTrace();  }  } catch (FileNotFoundException e) {  e.printStackTrace();  }finally{  if(reader != null)  try {  reader.close();  } catch (IOException e) {  // TODO Auto-generated catch block  e.printStackTrace();  }  }      }  } |

运行结果：

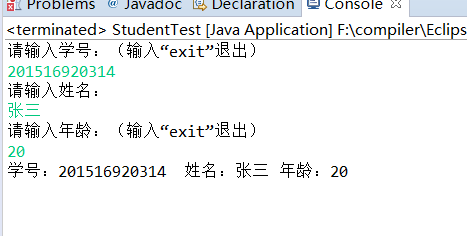


5 从控制台输入Student类的信息，包括学号、姓名、年龄，如输入错误，提示用户重新输入。创建该类，并在toString方法中显示个人信息。

代码：

|  |
| --- |
| package shiyan3;  import java.io.BufferedReader;  import java.io.IOException;  import java.io.InputStreamReader;  import java.util.Scanner;  public class StudentTest {  private long ID;  private String name;  private int age;  public StudentTest(long iD, String name, int age) {  super();  this.ID = iD;  this.name = name;  this.age = age;  }  public String toString(){  return "学号：" + this.ID + " 姓名：" + this.name + " 年龄：" + this.age;  }  public static void main(String[] args) {  long ID=0;  String name=null;  int age=0;  BufferedReader reader = null;  reader = new BufferedReader(new InputStreamReader(System.in));  String line = null;  try {  System.out.println("请输入学号：（输入“exit”退出）");  while((line = reader.readLine())!=null){  if(line.equalsIgnoreCase("exit"))  return;  try {  ID = Long.parseLong(line);  } catch (NumberFormatException e) {  System.out.println("学号格式错误");  }  if(ID != 0)  break;  }  System.out.println("请输入姓名：");  name = reader.readLine();  System.out.println("请输入年龄：（输入“exit”退出）");  while((line = reader.readLine())!=null){    if(line.equalsIgnoreCase("exit"))  return;  try {  age = Integer.parseInt(line);  } catch (NumberFormatException e) {  System.out.println("年龄格式错误");  }  if(age <0&& age > 150){  System.out.println("输入的年龄不正常");  continue;  }  if(age != 0) break;  }  } catch (IOException e) {  e.printStackTrace();  }  System.out.println(new StudentTest(ID,name,age).toString());  }  } |

运行结果：



三．在本次实验中的问题和解决方法

问题：在第1题的递归调用中总是出错。

解决方法：将文件和目录分开处理。

四．IV心得体会

这次试验的第一题，递归遍历目录。在递归的时候总是出错，经过分析，发现是因为当文件是目录的时候，调用错误。后来调用了File类中的listFiles()方法，返回该目录下的文件数组，然后循环遍历，就解决了这个问题。还有问题就是遍历显示后并不能显示出各个目录的层次。经过分析，便在前面添加了一个控制缩进的循环。这样显示后便有了层次感。

这次试验让我认识到了文件的相关操作，包括创建文件，遍历文件等。其次是输入流的应用，InputStream是字节输入流，Reader是字符输入流。还有输入流的相关嵌套可以添加各种方便的方法。