复习：

1. IO流：用于传输数据的API。

根据传输方向：输入流和输出力。

根据传输形式：字符流和字节流。

---InputStream OutputStream Reader Writer---都是抽象类。

数据的来源：硬盘、输入设备、内存、网络

文件字符流：操作字符类文件。字符输出流在输出的时候有缓冲区存在的。字符输入流没有缓冲区。

writer.close(); writer = null;

缓冲流：BufferdReader BufferedWriter 在创建对象的时候都需要传入一个流对象，实际读取数据靠的传入的流对象，缓冲流起的作用是提供一个缓冲区。---装饰者模式 --- Decorate Pattern

文件字节流：FileOutputStream, FileInputStream，在操作的时候没有缓冲区。

合并流：SequenceInputStream ，用于合并多个字节流的。在构建合并流对象的时候，需要传入Enumeration对象，利用了Vector中的elements()来获取这个Enumeration对象。

### 转换流

提供了字符流和字节流之间的相互转换。

OutputStreamWriter：将字符流转化为字节流。

InputStreamReader：将字节流转化为字符流

### 系统流/标准流

系统流都是字节流

System.in 标准输入流

System.out 标准输出流

System.err 标准错误流

练习：从控制台来读取一行数据

System.in -- InputStreamReader -- BufferedReader

### 打印流

PrintStream/PrintWriter--- 能够便捷的向外输出数据

### 序列**化/反序列化流**

将对象进行存储的过程就称之为序列化 --- 将对象保存到硬盘上就叫持久化

将对象还原回来的过程就称之为反序列化

注意：

1. 一个对象如果想要被序列化，那么要求这个对象对应的类必须实现一个接口---Serailizable --- 仅仅用于标志实现这个接口的类的对象允许被序列化

2. 被static/transient修饰的属性不能被序列化

3. 需要在类中手动指定一个属性---serialVersionUID---版本号。当一个对象序列化出去的时候会携带这个类的版本号，在反序列化的时候会检查对象携带的版本号和类中定义的版本号是否一致，如果一致则允许被反序列化。如果一个类中没有手动指定版本号，那么Java在编译的时候会根据当前类中的属性和方法自动计算产生一个版本号。也就意味着类中的属性或者方法产生变动，版本号就会改变。---因此需要手动指定，一旦手动指定就不再变化了。---版本号默认用private static final long 修饰限定

4. 在Java中数组和集合中的元素以及大部分的映射不能随着集合或者映射一起序列化

序列化和反序列化的加密机制---盒子---锁---钥匙---生产---了解即可

### Properties

是一个可以持久化的映射。键或者值只能是字符串。

必须要存储到.properties文件中---Properties文件的默认编码是西欧编码，因此不能存储中文---一旦放入中文就会出现乱码或者是转化为对应的Unicode编码

Properties经常作为配置文件使用

## JDK1.5的部分特性

自动封箱拆箱、增强for循环、静态导入、可变参数、枚举、泛型、反射、内省、动态代理、注解

### 静态导入/静态导包

import static 包名.类名.方法名; --- 表示导入指定包中的指定的静态方法

优点：能够一定程度的提高使用效率

缺点：书写不便；降低程序的可阅读性；如果在本类中存在了方法签名一致的静态方法，会导致静态导入无效

### 可变参数

用...定义一个可变参数---可变参数本质上就是一个数组，所以可以用操作数组的方式来操作一个可变参数。---可以传入任意多个参数也可以传入数组。---可以只传入一个参数也可以不传参数---对参数个数不做限定

可变参数只能定义一个，并且必须定义到参数列表的末尾。

### 枚举

取值情况相对固定并且能够一一列举---枚举

用enum定义枚举类。枚举常量之间用 , 隔开---枚举常量必须定义在枚举类的首行。

枚举类中的构造方法默认是私有的只能是私有的---构造方法可以重载

枚举类中可以定义一切的方法和属性，甚至包括抽象方法

在Java中，所有的枚举的顶级父类是java.lang.Enum

switch-case的()需要一个表达式，表达式的值可以是byte/short/char/int, jdk1.7允许使用String， jdk1.5开始允许使用枚举常量

### Junit测试---单元测试

不是Java原生提供的而是第三方厂商提供的现在被广泛应用的测试手段

@Test ---在要测试的方法上添加此注解 --- 在使用的时候要求 “三无”：无参无返回值非静态

### Debug---调试模式

断点调试

F6---执行到下一行

F5---钻入方法

F7---返回上一行

F8---执行到下一个断点