一、1-10题

1. 一个Java源程序文件中定义几个类和接口，则编译该文件后生成几个以.class为后缀的字节码文件。 （正确）

(1) 一个.java文件中定义多个类

① public权限类只能有一个（也可以一个都没有，但最多只有一个）；

② 这个.java文件名只能是public 权限的类的类名；

③ 倘若这个文件中没有public 类，则它的.java文件的名字是随便的一个类名；

④ 当用javac命令生成编译这个.java 文件的时候，则会针对每一个类生成一个.class文件；

2. 关于ASCII码和ANSI码，正确的特点。

(1) 标准ASCII只使用7个bit，扩展的ASCII使用8个bit。

(2) ANSI通常使用 0x00~0x7f 范围的1 个字节来表示 1 个英文字符。超出此范围的使用0x80~0xFFFF来编码，即扩展的ASCII编码。不同 ANSI 编码之间互不兼容。在简体中文Windows操作系统中，ANSI 编码代表 GBK 编码；在繁体中文Windows操作系统中，ANSI编码代表Big5；在日文Windows操作系统中，ANSI 编码代表 Shift\_JIS 编码。

(3) ANSI通常使用 0x00~0x7f 范围的1 个字节来表示 1 个英文字符，即ASCII码

(4) ASCII码包含一些特殊空字符，不全是打印字符。

4. 比较符“== 和 ！=”

==可用于基本类型和引用类型：当用于基本类型时候，是比较值是否相同；当用于引用类型的时候，是比较对象是否相同。

5. 多态正确的说法

(1) 使用父类类型的引用指向子类的对象；

(2) 该引用只能调用父类中定义的方法和变量；

(3) 如果子类中重写了父类中的一个方法，那么在调用这个方法的时候，将会调用子类中的这个方法；（动态连接、动态调用）

(4) 变量不能被重写（覆盖），”重写“的概念只针对方法，如果在子类中”重写“了父类中的变量，那么在编译时会报错。

(5) 不论向上或者向下转型，都是一句话，“**编译看左边，运行看右边**”。也就是编译时候，会看左边引用类型是否能正确编译通过，运行的时候是调用右边的对象的方法。

6. 执行下列Java代码 – 已整理

public class TestClass {

private static void testMethod(){

System.out.println("testMethod");

}

public static void main(String[] args) {

((TestClass)null).testMethod(); //null不是实例对象，可以转换成任意类型

}

}

(1) 方法是static静态方法，直接使用"类.方法"即可，因为静态方法使用不依赖对象是否被创建。null可以被强制类型转换成任意类型（不是任意类型对象），于是可以通过它来执行静态方法。

8. java用（监视器）机制实现了进程之间的同步执行

(1) 首先jvm中没有进程的概念，但是jvm中的线程映射为操作系统中的进程，对应关系为1：1。那这道题的问的就是jvm中线程如何异步执行.在jvm中 是使用监视器锁来实现不同线程的异步执行，在语法的表现就是synchronized 。

9. String str = new String(“abc”)，“abc”在内存中是怎么分配的？堆和字符串常量区

(1) 字符串常量池的概念 --已整理

JVM为了减少字符串对象的重复创建，其维护了一个特殊的内存，这段内存被成为字符串常量池或者字符串字面量池。

(2) 工作原理

**字符串常量池实现的前提条件就是Java中String对象是不可变的**，这样可以安全保证多个变量共享同一个对象。如果Java中的String对象可变的话，一个引用操作改变了对象的值，那么其他的变量也会受到影响，显然这样是不合理的。

(3) 堆栈的概念

Java中所有由类实例化的对象和数组都存放在堆内存中，无论是成员变量，局部变量，还是类变量，它们指向的对象都存储在堆内存中。而栈内存用来存储局部变量和方法调用。

(4) 寄存器概念

Java中运行时数据区有一个程序寄存器（又称程序计数器），该寄存器为线程私有。Java中的程序计数器用来记录当前线程中正在执行的指令。如果当前正在执行的方法是本地方法，那么此刻程序计数器的值为undefined。

10. Java对象的初始化方式（有待商榷）

(1) 构造器方法

(2) 通过class反射

实例化无参构造的类，先通过forName()方法实例化Class对象（即反射操作），然后用Class类中的newInstance()方法；实例化带参数构造的类，需要使用Class类中的getConstructors()方法获取所有的构造。

二、11-20题

1. 泛型知识语法糖，在编译后有无泛型都一样。

3. 数组的声明两种写法。

(1) int score[] = new int[10];

int[] score = new int[10];

4. Java中基本类型和引用类型

(1) 在Java中，变量有两种类型，一种是原始类型，一种是引用类型。

原始类型一共有8种，它们分别是char,boolean,byte,short,int,long,float,double。在Java API中，有它们对应的包装类，分别是(首字母大写)Character , Boolean , Byte , Short , Integer , Long , Float , Double(char,int的变化稍微大点)。

(2) JAVA JVM对于不同的原始类型会分配不同的存储空间，具体分配如下：

byte : 1个字节 8位 最大值: 127 (有符号)

short : 2个字节 16位 32767

int : 4个字节 32位 2147483647

long: 8个字节 64位 9223372036854775807

float: 4个字节 32位 3.4028235E38

double: 8个字节 64位 1.7976931348623157E308

(3) 枚举（enum）类型是Java 5新增的特性，它是一种新的类型，允许用常量来表示特定的数据片断，而且全部都以类型安全的形式来表示，是特殊的类，可以拥有成员变量和方法。

7. 以下是java concurrent包下的4个类，选出差别最大的一个：Future（异步）

(1) Semaphore：类，控制某个资源可被同时访问的个数;

(2) ReentrantLock：类，具有与使用synchronized方法和语句所访问的隐式监视器锁相同的一些基本行为和语义，但功能更强大；

(3) Future：接口，表示异步计算的结果；

(4) CountDownLatch： 类，可以用来在一个线程中等待多个线程完成任务的类。

9. Math.floor(-8.5)=( ) (double)-9.0

(1) floor: 求小于参数的最大整数。返回double类型

例如：Math.floor(-4.2) = -5.0

(2) ceil: 求大于参数的最小整数。返回double类型

例如：Math.ceil(5.6) = 6.0

(3) round: 对小数进行四舍五入后的结果。返回int类型

例如：Math.round(-4.6) = -5

10. Java中类的加载顺序 -- 已整理

(1) 父类静态代码块(包括静态初始化块，静态属性，但不包括静态方法)

(2) 子类静态代码块(包括静态初始化块，静态属性，但不包括静态方法 )

(3) 父类非静态代码块( 包括非静态初始化块，非静态属性 )

(4) 父类构造函数

(5) 子类非静态代码块 ( 包括非静态初始化块，非静态属性 )

(6) 子类构造函数

其中：类中静态块按照声明顺序执行，并且(1)和(2)不需要调用new类实例的时候就执行了(意思就是在类加载到方法区的时候执行的)

三、21-30题

2. System.out.printf()

(1) System是java.lang中的类，out为System中的一个静态成员，out是java.io.PrintStream类的对象，而println()是java.io.PrintStream类的方法，所以可以调用

类.静态成员.静态方法println()方法。

4. 执行下列代码：

public class Test{

private float f=1.0; //1.0默认为double，会发生double转float错误，应该加上f

int m=12;

static int n=1;

public static void main(String args[]){

Test t=new Test();

}

}

(1) 执行t.f会编译不通过，因为见注释。

(2) this的使用时针对在方法内部使局部变量等值于实例变量而使用的一个关键字，此处的n是静态变量而非实例变量 所以this的调用会出错。（不能用this调用静态变量）

5. 类的加载顺序，见二.10

8. 反射机制的作用（再去仔细了解）

(1) 在运行时判断任意一个对象所属的类；

(2) 在运行时构造任意一个类的对象；

(3) 在运行时判断任意一个类所具有的成员变量和方法；

(4) 在运行时调用任意一个对象的方法；生成动态代理。

9. 有关数组正确的说法

(1) 数组是一种派生类（引用类型），不是原生类（基本类型）。

(2) 数组大小一旦指定就不准再次改变。

四、31-40题

3. 内聚性和耦合性是度量软件模块独立性的重要准则，软件设计时应力求哪一个？

高内聚，低耦合。

4. 软件工程

(1) CMM共有五个等级，分别标志着软件企业能力成熟度的五个层次。从低到高，软件开发生产计划精度逐级升高，单位工程生产周期逐级缩短，单位工程成本逐级降低。据SEI统计，通过评估的软件公司对项目的估计与控制能力约提升40%到50%；生产率提高10%到20%，软件产品出错率下降超过1/3。

(2) 对一个软件企业来说，达到CMM2就基本上进入了规模开发，基本具备了一个现代化软件企业的基本架构和方法，具备了承接外包项目的能力。CMM3评估则需要对大软件集成的把握，包括整体架构的整合。

① 初始级

软件过程是无序的，有时甚至是混乱的，对过程几乎没有定义，成功取决于个人努力。管理是反应式的。

② 可管理级

建立了基本的项目管理过程来跟踪费用、进度和功能特性。制定了必要的过程纪律，能重复早先类似应用项目取得的成功经验。

③ 已定义级

已将软件管理和工程两方面的过程文档化、标准化，并综合成该组织的标准软件过程。所有项目均使用经批准、剪裁的标准软件过程来开发和维护软件，软件产品的生产在整个软件过程是可见的。

④ 量化管理级

分析对软件过程和产品质量的详细度量数据，对软件过程和产品都有定量的理解与控制。管理有一个作出结论的客观依据，管理能够在定量的范围内预测性能。

⑤ 优化管理级

过程的量化反馈和先进的新思想、新技术促使过程持续不断改进。

8. 软件开发模型

(1) 软件开发模型(Software Development Model)是指软件开发全部过程、活动和任务的结构框架。

(2) 1.边做边改模型2.瀑布模型3.演化模型4.增量模型5.螺旋模型6.喷泉模型

7.敏捷模型-SCRUM

(3) 瀑布模型 文档驱动 系统可能不满足客户的需求

快速原型模型 关注满足客户需求 可能导致系统设计差、效率低，难于维护

增量模型 开发早期反馈及时，易于维护 需要开放式体系结构，可能会设计差、效率低

螺旋模型 风险驱动 风险分析人员需要有经验且经过充分训练

五、41-50题

6. 形式参数

(1) 形式参数可被视为local variable（局部变量）。形参和局部变量一样都不能离开方法。都只有在方法内才会发生作用，也只有在方法中使用，不会在方法外可见。

(2) **对于形式参数只能用final修饰符，其它任何修饰符都会引起编译器错误。**但是用这个修饰符也有一定的限制，就是在方法中不能对参数做任何修改。 不过一般情况下，一个方法的形参不用final修饰。只有在特殊情况下，那就是：方法内部类。 一个方法内的内部类如果使用了这个方法的参数或者局部变量的话，这个参数或局部变量应该是final。

(3) 方法的参数列表指定要传递给方法什么样的信息，采用的都是对象的形式。引用传递。

8. 变量a是一个64位有符号的整数，初始值用16进制表示为：0x7FFFFFFFFFFFFFFF;变量b是一个64位有符号的整数，初始值用16进制表示为：0x8000000000000000。则a+b的结果用10进制表示为多少？

(1) a+b=0xFFFFFFFFFFFFFFFF，十六进制的F对应十进制的15，再转换成二进制则为1111。

所以a+b得1111 1111 1111 1111，二进制的首尾为符号位，1位负数，0则为正数，这里为1所以a+b的结果是负数。

(2) 负数在内存中存储以补码的形式，所以求原码的步骤为：

1.补码-1得反码（变为1111 1111 1111 1110）

2.反码按位取反得原码（0000 0000 0000 0001）

3.转换成二进制结果为-1

9. byte b1=1,b2=2,b3,b6;

final byte b4=4,b5=6;

b6=b4+b5;

b3=(b1+b2);

System.out.println(b3+b6);

(1) 被final修饰的变量是常量，这里的b6=b4+b5可以看成是b6=10；在编译时就已经变为b6=10了

(2) 而b1和b2是byte类型，java中进行计算时候将他们提升为int类型，再进行计算，b1+b2计算后已经是int类型，赋值给b3，b3是byte类型，类型不匹配，编译不会通过，需要进行强制转换。

Java中的byte，short，char进行计算时都会提升为int类型。

六、51-60题

8. 代码String str=”123456a”；int i=Integer.parseInt(str);会报异常的是（）

java.lang.NumberFormatException -- 已整理

(1) 非纯数字的字符串转化为Integer对象会报数字格式异常。

七、61-80题

1. 抽象类和接口的默认访问权限

(1) 关于抽象类

JDK 1.8以前，抽象类的方法默认访问权限为protected

JDK 1.8时，抽象类的方法默认访问权限变为default

(2) 关于接口

JDK 1.8以前，接口中的方法必须是public的

JDK 1.8时，接口中的方法可以是public的，也可以是default的

JDK 1.9时，接口中的方法可以是private的

2. java指令

(1) jar 将许多文件组合成一个jar文件

(2) javac 编译

(3) javadoc 它从程序源代码中抽取类、方法、成员等注释形成一个和源代码配套的API帮助文档。

(4) javah 把java代码声明的JNI方法转化成C\C++头文件。 JNI可参考java核心技术卷二第12章

3. 异常体系try-catch-finally

(1) 先执行try，不管有没有执行catch都会执行finally。如果try中有return，则先执行finally再执行return。

5. 执行下列代码： -- 已整理

public class SendValue{

public String str="6";

public static void main(String[] args) {

SendValue sv=new SendValue();

sv.change(sv.str);

System.out.println(sv.str);

}

public void change(String str) {

str="10";

}

} 输出：6

(1) 值传递和引用传递的区别

① 值传递：(形式参数类型是基本数据类型)：方法调用时，实际参数把它的值传递给对应的形式参数，**形式参数只是用实际参数的值初始化自己的存储单元内容**，是两个不同的存储单元，**所以方法执行中形式参数值的改变不影响实际参数的值。**

② 引用传递：(形式参数类型是引用数据类型参数)：也称为传地址。方法调用时，实际参数是对象(或数组)，这时实际参数与形式参数指向同一个地址，在方法执行中，对形式参数的操作实际上就是对实际参数的操作，这个结果在方法结束后被保留了下来，所以**方法执行中形式参数的改变将会影响实际参数。**

10. Java数据库连接库JDBC用到哪种设计模式?

(1) 桥接模式

JDBC连接 数据库 的时候，在各个数据库之间进行切换，基本不需要动太多的代码，甚至丝毫不动，原因就是JDBC提供了统一接口，每个数据库提供各自的实现，用一个叫做数据库驱动的程序来桥接就行了

八、71-80题

1. 数组复制方法效率比较

System.arraycopy>clone>Arrays.copyOf>for循环

3. 异常体系

非RuntimeException，非Error，也包括IOException



4. Volatile关键字

(1) 出于运行速率的考虑，java编译器会把经常经常访问的变量放到缓存（严格讲应该是工作内存）中，读取变量则从缓存中读。但是在多线程编程中,内存中的值和缓存中的值可能会出现不一致。volatile用于限定变量只能从内存中读取，保证对所有线程而言，值都是一致的。但是volatile不能保证原子性，也就不能保证线程安全。

5. 结构模式（设计模式）

结构型模式是描述如何将类对象结合在一起，形成一个更大的结构，结构模式描述两种不同的东西：类与类的实例。故可以分为类结构模式和对象结构模式。

6. 字节流和字符流

(1) 字节流：

InputStream

|-- FileInputStream (基本文件流）

|-- BufferedInputStream

|-- DataInputStream

|-- ObjectInputStream

(2) 字符流

① Reader

|-- InputStreamReader (byte->char 桥梁）

|-- BufferedReader (常用）

② Writer

|-- OutputStreamWriter (char->byte 桥梁）

|-- BufferedWriter

|-- PrintWriter （常用）

9. finalize()方法

方法名。Java 技术允许使用 finalize() 方法在垃圾收集器将对象从内存中清除出去之前做必要的清理工作。这个方法是由垃圾收集器在确定这个对象没有被引用时对这个对象调用的。它是在 Object 类中定义的，因此所有的类都继承了它。**子类覆盖 finalize() 方法以整理系统资源或者执行其他清理工作**。finalize() 方法是在垃圾收集器删除对象之前对这个对象调用的。注意：finalize不一定被jvm调用，只有当垃圾回收器要清除垃圾时才被调用。

10. 异常关键字的使用

(1) throws用于在方法上声明该方法不需要处理的异常类型,用在方法上后面跟异常类名 可以是多个异常类

(2) throw用于抛出具体异常类的对象,用在方法内 后面跟异常对象只能是一个异常类型实体.

(3) try块必须和catch块或和finally同在,不能单独存在,二者必须出现一个.

(4) finally块总会执行,不论是否有错误出现.但是若try语句块或会执行的catch语句块使用了JVM系统退出语句,finally块就不会被执行了. 一般我们把关闭资源的代码放在finally里面 保证资源总是能关闭

九、81-90

4. 实例变量声明有默认值，局部变量声明没有默认值。

boolean类型的默认值是false。

5. 子类不能访问父类中private修饰的成员。

6. 如何获取ServletContext设置的参数值？getInitParameter

(1) getParameter()是获取POST/GET传递的参数值；

(2) getInitParameter获取Tomcat的server.xml中设置Context的初始化参数

(3) getAttribute()是获取对象容器中的数据值；

(4) getRequestDispatcher是请求转发。

8. 下列是正确的说法

(1) ConcurrentHashMap使用segment来分段和管理锁，segment继承自ReentrantLock，因此ConcurrentHashMap使用ReentrantLock来保证线程安全。

(2) Arrays.asList()

将一个数组转化为一个List对象，这个方法会返回一个ArrayList类型的对象， 这个ArrayList类并非java.util.ArrayList类，而是Arrays类的静态内部类！用这个对象对列表进行添加删除更新操作，就会报UnsupportedOperationException异常。

(3) SimpleDateFormat对象是线程不安全的。

10. 二维数组的声明方式

数组命名时名称与[]可以随意排列，但声明的二维数组中第一个中括号中必须要有值，它代表的是在该二维数组中有多少个一维数组。

(1) float f[][] = new float[6][6];

(2) float []f[] = new float[6][6];

(3) float [][]f = new float[6][6];

(4) float [][]f = new float[6][];

十、91-100题

2. 输入流将数据从文件，标准输入或其他外部输入设备中加载道内存，在 java 中其对应于抽象类（）及其子类。

java.io.InputStream 输入（到程序）

java.io.OutputStream （从程序）输出

6. java类中的方法可以和类名同名（但是一般不推荐这样做）

7. HttpServletRequest类主要处理：（考察HttpServletRequest）

(1) 读取和写入HTTP头标

(2) 取得和设置cookies

(3) 取得路径信息

(4) 标识HTTP会话

10. 静态初始化块 -- 已整理

(1) 无法直接调用静态初始化块。

(2) 在创建第一个实例前或引用任何静态成员之前，将自动调用静态初始化块来初始化。

(3) 静态初始化块既没有访问修饰符，也没有参数。

十一、101-110题

2. 一个文件中的数据要在控制台上显示，首先需要（ ）。

//新建一个文件流

FileInputStream fin = new FileInputStream(this.filename);。

3. 算法概念

(1) 有穷性（Finiteness）

算法的有穷性是指算法必须能在执行有限个步骤之后终止；

(2) 确切性(Definiteness)

算法的每一步骤必须有确切的定义；

(3) 输入项(Input)

一个算法有0个或多个输入，以刻画运算对象的初始情况，所谓0个输入是指算法本身定出了初始条件；

(4) 输出项(Output)

一个算法有一个或多个输出，以反映对输入数据加工后的结果。没有输出的算法是毫无意义的；

(5) 可行性(Effectiveness)

算法中执行的任何计算步骤都是可以被分解为基本的可执行的操作步，即每个计算步都可以在有限时间内完成（也称之为有效性）。

10.获取JSP中定义的对象

https://uploadfiles.nowcoder.net/images/20161123/5918115_1479891741965_E3A349EA0FAFB7A49B99084AB8BF7860

(1) 通过 pageContext（PageContext类的实例，提供对JSP页面所有对象以及命名空间的访问） 获取stringBean 的值，赋给 String 类型的变量 myBean，然后通过 Jsp 表达式 处理该值。

<% String myBean =

(String)pageContext.getAttribute(“stringBean”,PageContext.PAGE\_SCOPE);%>

<%=myBean%>

(2) bean:write相当于 <%=request.getAttribute("something")%> 其中 something 是属性的名字

<bean:write name=“stringBean”/>

(3) 通过Jsp 表达式 获取变量 stringBean 的值

<%=stringBean%>

十二、111-120题

1. char类型的字符都可以转换成int类型数字进行运算

5. JDK中提供的java、javac、jar等开发工具也是用Java编写的。

9. Web Service（WSDL 可描述网络服务）

(1) Web service顾名思义是基于web的服务，它是一种跨平台，跨语言的服务。

(2) WSDL 是一种使用 XML 编写的文档。这种文档可描述某个 Web service。它可规定服务的位置，以及此服务提供的操作（或方法）

(3) 采用了soap协议（简单对象协议）进行通信

十三、121-130题

1. package语句必须作为源文件的第一条非注释性语句，一个源文件只能指定一个包，只能包含一条package语句。

2. Java三大注解

(1) @SuppressWarnings(“deprecation”)的功能是屏蔽不赞同（就是过时废弃的意思）使用的类和方法的警告

(2) Suppresswarnings 注解告诉编译器应该屏蔽带注解的元素和所有子元素的警告信息. 会压制一个元素集和子元素的所有警告信息.

(3) @Override重写的注解

9. 初始化过程 -- 已整理

(1) 初始化父类中的静态成员变量和静态代码块 ；

(2) 初始化子类中的静态成员变量和静态代码块 ；

(3) 初始化父类的普通成员变量和代码块，再执行父类的构造方法；

(4) 初始化子类的普通成员变量和代码块，再执行子类的构造方法；

10. try-catch-finally块中，finally块在以下几种情况将不会执行。

(1) finally块中发生了异常。

(2) 程序所在线程死亡。

(3) 在前面的代码中用了System.exit（）；

(4) 关闭了CPU

十四、131-140题

7. 内存使用

(1) 堆区：只存放类对象，线程共享；

(2) 方法区：又叫静态存储区，存放class文件和静态数据，线程共享;

(3) 栈区：存放方法局部变量，基本类型变量区、执行环境上下文、操作指令区，线程不共享

8. Hibernate相关

优化Hibernate所鼓励的7大措施：

(1) 尽量使用many-to-one，避免使用单项one-to-many

(2) 灵活使用单向one-to-many

(3) 不用一对一，使用多对一代替一对一

(4) 配置对象缓存，不使用集合缓存

(5) 一对多使用Bag 多对一使用Set

(6) 继承使用显示多态 HQL:from object polymorphism="exlicit" 避免查处所有对象

(7) 消除大表，使用二级缓存

10. GC回收

(1) 两个最基本的java回收算法：复制算法和标记清理算法

① 复制算法：两个区域A和B，初始对象在A，继续存活的对象被转移到B。此为新生代最常用的算法

② 标记清理：一块区域，标记可达对象（可达性分析），然后回收不可达对象，会出现碎片，那么引出标记-整理算法：多了碎片整理，整理出更大的内存放更大的对象

(2) 两个概念：新生代和年老代

新生代：初始对象，生命周期短的

永久代：长时间存在的对象

整个java的垃圾回收是新生代和年老代的协作，这种叫做分代回收。

P.S：Serial New收集器是针对新生代的收集器，采用的是复制算法

Parallel New（并行）收集器，新生代采用复制算法，老年代采用标记整理

Parallel Scavenge（并行）收集器，针对新生代，采用复制收集算法

Serial Old（串行）收集器，新生代采用复制，老年代采用标记整理

Parallel Old（并行）收集器，针对老年代，标记整理

CMS收集器，基于标记清理

G1收集器：整体上是基于标记 整理 ，局部采用复制

综上：新生代基本采用复制算法，老年代采用标记整理算法。cms采用标记清理。

十五、141-150题

5. 获得Servlet的初始化参数

通过ServletConfig接口的getInitParameter(java.lang.String name)方法

十六、151-160题

3. 变量的命名规则：标识符可以包括这4种字符：字母、下划线、$、数字；开头不能是数字；不能是关键字

4. JDBC

一般关系数据模型和对象数据模型之间有以下对应关系：表对应类，记录对应对象，表的字段对应类的属性

5. 中间件

中间件是一种独立的系统软件或服务程序，分布式应用软件借助这种软件在不同的技术之间共享资源。中间件位于客户机/ 服务器的操作系统之上，管理计算机资源和网络通讯。是连接两个独立应用程序或独立系统的软件。相连接的系统，即使它们具有不同的接口，但通过中间件相互之间仍能交换信息。执行中间件的一个关键途径是信息传递。通过中间件，应用程序可以工作于多平台或OS环境。

（简单来说，中间件并不能提高内核的效率，一般只是负责网络信息的分发处理）

6. Java并发

(1) CopyOnWriteArrayList适用于写少读多的并发场景

(2) ReadWriteLock即为读写锁，他要求写与写之间互斥，读与写之间互斥，

读与读之间可以并发执行。在读多写少的情况下可以提高效率

(3) ConcurrentHashMap是同步的HashMap，读写都加锁

(4) volatile只保证多线程操作的可见性，不保证原子性

7. 执行下列代码

public class Test2

{

public void add(Byte b)

{

b = b++;

}

public void test()

{

Byte a = 127;

Byte b = 127;

//在进入add前，a会自增1。Byte类型值大小为-128~127之间，所以会越界，编程-128

add(++a); //add函数并不会改变a和b的值，所以a=-128，b=127

System.out.print(a + " ");

add(b);

System.out.print(b + "");

}

}

8. 关于静态方法 -- 已整理

(1) 在类方法中不能用this来调用本类的类方法。

(2) 在类方法中调用本类的类方法时可直接调用。

9. 类的修饰符

(1) 因为普通类也就是外部类，通过 eclipse 的警告“Illegal modifier for the class Test; only public, abstract & final are permitted” 可知只能用 public, abstract 和 final 修饰。

(2) 内部类则可以用 修饰成员变量的修饰符修饰内部类，比如 private, static, protected 修饰。

10. 类型的自动转换

数据类型的转换，分为自动转换和强制转换。自动转换是程序在执行过程中 “ 悄然 ” 进行的转换，不需要用户提前声明，一般是从位数低的类型向位数高的类型转换；强制类型转换则必须在代码中声明，转换顺序不受限制。

(1) 自动数据类型转换

自动转换按从低到高的顺序转换。不同类型数据间的优先关系如下：

低 ---------------------------------------------> 高

byte,short,char-> int -> long -> float -> double

十七、161-170题

6. java 的字符类型采用的是 Unicode 编码方案

注意：区分编码和编码格式

(1) 编码： 编码就是一个编号(数字)到字符的一种映射关系，就仅仅是一种一对一的映射而已，可以理解成一个很大的对应表格。 java默认的字符集是Unicode（占两个字节byte，一个字节=8比特位bit，所以每个Unicode占用16比特位）

(2) 编码格式：编码格式 是用来序列化或存储编码中提到的那个“编号(数字)”的一种“格式”，包括gbk和utf-8

gbk： 是指中国的中文字符，其它它包含了简体中文与繁体中文字符

UTF-8： 它是一种全国家通过的一种编码

9. 接口中可以不声明任何方法，和成员变量

抽象类可以不包含抽象方法，但有抽象方法的类一定要声明为抽象类

十八、171-180题

4. 存根（Stub）与以下哪种技术有关 ：动态链接

存根类是一个类，它实现了一个接口，它的作用是：如果一个接口有很多方法，如果要实现这个接口，就要实现所有的方法。但是一个类从业务来说，可能只需要其中一两个方法。如果直接去实现这个接口，除了实现所需的方法，还要实现其他所有的无关方法。而如果通过继承存根类就实现接口，就免去了这种麻烦。

8. 接口

接口中方法默认是 abstract public,所以在接口只写函数声明是符合语法规则。但是变量默认是用public final static 修饰的，意思它是静态常量，常量不管在接口中还是类中必须在声明时初始化！

十九、181-190题

1. 常见字符的ASCII码值如下：空格的ASCII码值为32；数字0到9的ASCII码值分别为48到57；大写字母“A”到“Z”的ASCII码值分别为65到90；小写字母“a”到“z”的ASCII码值分别为97到到122。

2. 关于管道

管道实际上是一种固定大小的缓冲区，管道对于管道两端的进程而言，就是一个文件，但它不是普通的文件，它不属于某种文件系统，而是自立门户，单独构成一种文件系统，并且只存在于内存中。它类似于通信中半双工信道的进程通信机制，一个管道可以实现双向 的数据传输，而同一个时刻只能最多有一个方向的传输，不能两个方向同时进行。管道的容 量大小通常为内存上的一页，它的大小并不是受磁盘容量大小的限制。当管道满时，进程在 写管道会被阻塞，而当管道空时，进程读管道会被阻塞。

匿名管道只能单向；命名管道可以双向。

8. HashMap和HashTable区别

(1) Hashtable和HashMap的区别主要是前者是同步的，后者是快速失败机制保证。

(2) HashTable不允许null值(key和value都不可以),HashMap允许null值(key和value都可以)。

9. java中的关键字，保留字

java中true ,false , null在java中不是关键字，也不是保留字，它们只是显式常量值，但是你在程序中不能使用它们作为标识符。其中const和goto是java的保留字。java中所有的关键字都是小写的，还有要注意true,false,null,　friendly，sizeof不是java的关键字,但是你不能把它们作为java标识符用。

二十、191-200题

4. JVM-GC垃圾回收是作用于堆上的，不用于局部变量（栈上）

java提供了一个系统级的线程，即垃圾回收器线程。用来对每一个分配出去的内存空间进行跟踪。当JVM空闲时，自动回收每块可能被回收的内存，GC是完全自动的，不能被强制执行。程序员最多只能用System.gc()来建议执行垃圾回收器回收内存，但是具体的回收时间，是不可知的。

5. JSP 四大作用域： page (作用范围最小)、request、session、application（作用范围最大）。

(1) 存储在application对象中的属性可以被同一个WEB应用程序中的所有Servlet和JSP页面访问。（属性作用范围最大）

(2) 存储在session对象中的属性可以被属于同一个会话（浏览器打开直到关闭称为一次会话，且在此期间会话不失效）的所有Servlet和JSP页面访问。

(3) 存储在request对象中的属性可以被属于同一个请求的所有Servlet和JSP页面访问（在有转发的情况下可以跨页面获取属性值），例如使用PageContext.forward和PageContext.include方法连接起来的多个Servlet和JSP页面。

(4) 存储在pageContext对象中的属性仅可以被当前JSP页面的当前响应过程中调用的各个组件访问，例如，正在响应当前请求的JSP页面和它调用的各个自定义标签类。

6. final关键字

final修饰的成员变量在赋值时可以有三种方式。1、在声明时直接赋值。2、在构造器中赋值。3、在初始代码块中进行赋值。

9. SpringMVC

SpringMVC是Spring中的模块，它实现了mvc设计模式的web框架，首先用户发出请求，请求到达SpringMVC的前端控制器（DispatcherServlet）,前端控制器根据用户的url请求处理器映射器查找匹配该url的handler，并返回一个执行链，前端控制器再请求处理器适配器调用相应的handler进行处理并返回给前端控制器一个modelAndView，前端控制器再请求视图解析器对返回的逻辑视图进行解析，最后前端控制器将返回的视图进行渲染并把数据装入到request域，返回给用户。