1. 定义有StringBuffer s1=new StringBuffer(10);s1.append(“1234”)则s1.length()和s1.capacity()分别是多少?

(1) length 返回当前长度，capacity返回初始化的长度，如果有初始化返回初始化，默认为16。

(2) StringBuffer和StringBuilder的默认大小为16，ArrayList和LinkedList的默认大小10。

2. 线程的方法

resume与suspended一起使用

wait与notify(notifyAll)一起使用

sleep会让线程暂时不执行

3. String类相关 -- 已整理

(1) new String("zz")实际上创建了2个String对象，就是使用“zz”通过双引号创建的（在字符串常量池），另一个是通过new创建的（在堆里）。只不过他们的创建的时期不同，一个是编译期，一个是运行期。

(2) String str1="hello"; //双引号在编译期确定，新建对象在运行期确定

String str2="he"+ new String("llo"); //当String str2 = “he”+”llo”时，判断为true

System.out.println(str1==str2); //判断为false

4. java编译相关

语言跨平台是编译后的文件跨平台，而不是源程序跨平台。Java源代码首先经过编译器生成字节码，即class文件，该class文件与平台无关，而class文件经过解释执行之后翻译成最终的机器码，这是平台相关的。

5. Lanbda表达式（Java8）

Lanbda表达式的主要作用就是代替匿名内部类的繁琐语法， 它由三部分组成：

(1) 形参列表。形参列表允许省略形参类型。如果形参列表中只有一个参数，甚至连形参列表的圆括号也可以省略。

(2) 箭头（→）。必须通过英文中画线和大于符号组成。

(3) 代码块。如果代码块只包含一条语句，Lambda表达式允许省略代码块的花括号，那么那条语句就不要用花括号表示语句结束。Lambda代码块只有一条return语句，甚至可以省略return关键字。Lambda表达式需要返回值，而它的代码块中仅有一套省略了return的语句。Lambda表达式会自动返回这条语句的值。

实例： dir.listFiles((File f)->f.getName().endsWith(“.Java”));

dir.listFiles( f->f.getName().endsWith(“.Java”));

6. Java事件处理模型三要素

(1) 事件源（Event Source）：即事件发生的场所，就是指各个组件，如按钮等，点击按钮其实就是组件上发生的一个事件；

(2) 事件（Event）：事件封装了组件上发生的事情，比如按钮单击、按钮松开等等；

(3) 事件监听器（Event Listener）：负责监听事件源上发生的特定类型的事件，当事件到来时还必须负责处理相应的事件；

7. Java接口的修饰符

(1) 接口中的属性的默认是public static final //不能为private和protected

(2) 方法是public abstract //不能为private和protected

(3) 类的成员变量或成员方法声明为默认权限，则其是包级可见，同一个包内的类可以访问到这个属性或方法friendly

8. 关于Cookie和Session

(1) Cookie是Web服务器发送给客户端的一小段信息，客户端请求时，可以读取该信息发送到服务器端，包含了sessionID。

(2) 关闭浏览器意味着临时会话ID丢失，但所有与原会话关联的会话数据仍保留在服务器上，直至会话过期。

(3) 在禁用Cookie时可以使用URL重写技术跟踪会话

(4) 隐藏域在页面中对于用户（浏览器）是不可见的，在表单中插入隐藏域的目的在于收集或发送信息，以利于被处理表单的程序所使用。浏览者单击发送按钮发送表单的时候，隐藏域的信息也被一起发送到服务器。

9. volatile关键字

一旦一个共享变量（类的成员变量、类的静态成员变量）被volatile修饰之后，那么就具备了两层语义：

(1) 保证了不同线程对这个变量进行操作时的可见性，即一个线程修改了某个变量的值，这新值对其他线程来说是立即可见的。

(2) 禁止进行指令重排序。

(3) volatile并不提供原子性的保证，多线程下计数器必须使用锁保护。

10. 自动转换类型

两个数值进行二元操作时，会有如下的转换操作：

如果两个操作数其中有一个是double类型，另一个操作就会转换为double类型。

否则，如果其中一个操作数是float类型，另一个将会转换为float类型。

否则，如果其中一个操作数是long类型，另一个会转换为long类型。

否则，两个操作数都转换为int类型。

11. Java程序的种类

(1) 内嵌于Web文件中，由浏览器来观看的\_Applet

(2) 可独立运行的 Application

(3) 服务器端的 Servlets

12. Java编译和运行的命令

(1) 运行命令：java + Java 程序的名字但是不加后缀

(2) 编译命令：javac + Java 程序的名字加上“.java”后缀

13. Switch参数

switch语句后的控制表达式只能是short、char、int、long整数类型和枚举类型，不能是float，double和boolean类型。String类型是java7开始支持。

14. 接口

(1) 类只能继承一个类，可以实现多个接口。

(2) 接口可以继承（或扩展）多个接口。

15. Java中的jar包

java.awt： 包含构成抽象窗口工具集的多个类，用来构建和管理应用程序的图形用户界面

java.lang： 提供java编成语言的程序设计的基础类

java.io：　 包含提供多种输出输入功能的类，

java.net：　 包含执行与网络有关的类，如URL，SCOKET，SEVERSOCKET，

java.applet： 包含java小应用程序的类

java.util：　 包含一些实用性的类

16. Java的堆内存

(1) java的堆内存分为两块:permantspace（持久带） 和 heap space。

持久带中主要存放用于存放静态类型数据，如 Java Class, Method 等， 与垃圾收集器要收集的Java对象关系不大。而heapspace分为年轻带和年老带。

(2) 年轻代的垃圾回收叫 Young GC， 年老代的垃圾回收叫 Full GC。

在年轻代中经历了N次（可配置）垃圾回收后仍然存活的对象，就会被复制到年老代中。因此，可以认为年老代中存放的都是一些生命周期较长的对象

年老代溢出原因：循环上万次的字符串处理、创建上千万个对象、在一段代码内申请上百M甚至上G的内存。

持久代溢出原因：动态加载了大量Java类而导致溢出

17. 类的修饰符（也可以不写修饰符）

类的修饰符分为：可访问控制符和非访问控制符两种。

可访问控制符是: 公共类修饰符 public //不能使用private和protected

非访问控制符有：抽象类修饰符 abstract

最终类修饰符：final