JAVA 面试 32 问

第一，谈谈 final，finally，finalize 的区别。 （最常被问到）

final 修饰符（关键字）
如果一个类被声明为 final，意味着它不能再派生出新的子类，不能作为父类被继承。因此一个类不能既被声明为 abstract，又被声明为 final。将变量或方法声明为 final，可以保证它们在使用中不被改变。其初始化可以在两个地方：一是其定义处，也就是说在 final 变量定义时直接给其赋值；二是在构造函数中。这两个地方只能选其一，要么在定义时给值，要么在构造函数中给值，不能同时既在定义时给了值，又在构造函数中给另外的值，而在以后的引用中只能读取，不可修改。被声明为 final 的方法也同样只能使用，不能重写（override）。
finally
在异常处理时提供 finally 块来执行任何清除操作。如果抛出一个异常，那么相匹配的 catch 子句就会执行，然后控制就会进入 finally 块（如果有的话）。
finalize
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。

第二，Anonymous Inner Class（匿名内部类） 是否可以 extends（继承）其它类，是否可以 implements（实现）interface（接口）？

匿名的内部类是没有名字的内部类。不能 extends（继承） 其它类，但一个内部类可以作为一个接口，由另一个内部类实现。

第三，Static Nested Class 和 Inner Class 的不同，说得越多越好。（面试题有的很笼统）

Nested（嵌套）Class（一般是 C++的说法）
nested class 是合成型聚集关系（Composite Aggregation） 的另一种表达方式，也就是说 nested class 也可以用 Aggregation 表达出来，但是，nested class 更加精确地表达了一种专用的、紧耦合的关系，尤其在代码生成时，nested class 在 Java 中映射成 inline class。比如，计算机专用开关电源类可以作为计算机类的 nested class，但是，电池组类就不一定适合作为计算机类的 nested class，因为，电池组类表述的是一个过于通用的对象，可能还被包含（Aggregation）于模型中的其他设备对象。class A nested in class B，则说明 A 是一个 nested class，一般 A 是用来完成 B 中的某种重要功能。
Inner Class (一般是 JAVA 的说法)
Java 内部类与 C++嵌套类最大的不同就在于是否有指向外部的引用上。

静态内部类（Inner Class）意味着 a 创建一个 static 内部类的对象，不需要一个外部类对象，b 不能从一个 static 内部类的一个对象访问一个外部类对象

第四，&和&&的区别。（这个问得很少）

&是位运算符。&可为位运算，也可为逻辑运算，看情况而定。&&是布尔逻辑运算符。

第五，HashMap 和 Hashtable 的区别。（常问）

都属于 Map 接口的类，实现了将惟一键映射到特定的值上。
HashMap 类没有分类或者排序。它允许一个 null 键和多个 null 值。

Hashtable 类似于 HashMap，但是不允许 null 键和 null 值。它也比 HashMap 慢，因为它是同步的。
Hashtable 继承自 Dictionary 类，而 HashMap 是 Java1.2 引进的 Map interface 的一个实现。
HashMap 允许将 null 作为一个 entry 的 key 或者 value，而 Hashtable 不允许，还有就是，HashMap 把 Hashtable 的 contains 方法去掉了，改成 containsvalue(Returns true if this map maps one or more keys to the specified value)和 containsKey(Returns true if this map contains a mapping for the specified key)。因为 contains(Tests if some key maps into the specified value in this hashtable)方法容易让人引起误解。
最大的不同是，Hashtable 的方法是 Synchronize 的，而 HashMap 不是，在多个线程访问 Hashtable 时，不需要自己为它的方法实现同步，而 HashMap 就必须为之提供外同步。
Hashtable 和 HashMap 采用的 hash/rehash 算法都大概一样，所以性能不会有很大的差异。

第六，Collection 和 Collections 的区别。（你千万别说一个是单数一个是复数）

Collections 是个 java.util 下的类，它包含有各种有关集合操作的静态方法。
Collection 是个 java.util 下的接口，它是各种集合结构的父接口。

第七，什么时候用 assert。（API 级的技术人员有可能会问这个）

断言是一个包含布尔表达式的语句，在执行这个语句时假定该表达式为 true。如果表达式计算为 false，那么系统会报告一个 Assertionerror。它用于调试目的：
assert(a > 0); // throws an Assertionerror if a <= 0

断言可以有两种形式：
assert Expression1 ;
assert Expression1 : Expression2 ;
Expression1 应该总是产生一个布尔值。
Expression2 可以是得出一个值的任意表达式。这个值用于生成显示更多调试信息的 String 消息。
断言在默认情况下是禁用的。要在编译时启用断言，需要使用 source 1.4 标记：
javac -source 1.4 Test.java
要在运行时启用断言，可使用 -enableassertions 或者 -ea 标记。
要在运行时选择禁用断言，可使用 -da 或者 -disableassertions 标记。
要系统类中启用断言，可使用 -esa 或者 -dsa 标记。还可以在包的基础上启用或者禁用断言。
可以在预计正常情况下不会到达的任何位置上放置断言。断言可以用于验证传递给私有方法的参数。不过，断言不应该用于验证传递给公有方法的参数，因为不管是否启用了断言，公有方法都必须检查其参数。不过，既可以在公有方法中，也可以在非公有方法中利用断言测试后置条件。另外，断言不应该以任何方式改变程序的状态。

第八，GC 是什么？为什么要有 GC？（基础）

GC 是垃圾收集器。Java 程序员不用担心内存管理，因为垃圾收集器会自动进行管理。要请求垃圾收集，可以调用下面的方法之一：
System.gc()
Runtime.getRuntime().gc()
gc 虽可清理，但非立刻清理

第九，String s = new String("xyz");创建了几个 String Object？

两个对象，一个是"xyx",一个是指向"xyx"的引用对象 s。一个对象，一个对象引用。

第十，Math.round(11.5)等于多少？ Math.round(-11.5)等于多少？

Math.round(11.5)返回（long）12，Math.round(-11.5)返回（long）-11;

第十一，short s1 = 1; s1 = s1 + 1;有什么错？ short s1 = 1; s1 += 1;有什么错？ （ 面试题都是很变态的，要做好受虐的准备）

short s1 = 1; s1 = s1 + 1;有错，s1 是 short 型，s1+1 是 int 型,不能显式转化为 short 型。可修改为 s1 =(short)(s1 + 1) 。short s1 = 1; s1 += 1 正确。

Byte a=1,b=1,bye c=a+b;错误，byte 与 byte，short 与 short 运算返回值为 int

第十二，sleep() 和 wait() 有什么区别？ （搞线程的最爱）

sleep()
是使线程停止一段时间的方法。在 sleep 时间间隔期满后，线程不一定立即恢复执行。这是因为在那个时刻，其它线程可能正在运行而且没有被调度为放弃执行，除非(a)"醒来"的线程具有更高的优先级 (b)正在运行的线程因为其它原因而阻塞。
wait()
是线程交互时，如果线程对一个同步对象 x 发出一个 wait()调用，该线程会暂停执行，被调对象进入等待状态，直到被唤醒或等待时间到。

第十三，Java 有没有 goto？（很十三的问题，如果哪个面试的问到这个问题，我劝你还是别进这家公司）

Goto java 中的保留字，现在没有在 java 中使用。

第十四，数组有没有 length()这个方法？ String 有没有 length()这个方法？

数组没有 length()这个方法，有 length 的属性。
String 有有 length()这个方法。

第十五，Overload 和 Override 的区别。Overloaded 的方法是否可以改变返回值的类型？ （常问）

方法的重写 Overriding 和重载 Overloading 是 Java 多态性的不同表现。重写 Overriding 是父类与子类之间多态性的一种表现，重载 Overloading 是一个类中多态性的一种表现。如果在子类中定义某方法与其父类有相同的名称和参数，我们说该方法被重写 (Overriding)。子类的对象使用这个方法时，将调用子类中的定义，对它而言，父类中的定义如同被"屏蔽"了。如果在一个类中定义了多个同名的方法，它们或有不同的参数个数或有不同的参数类型，则称为方法的重载(Overloading)。Overloaded 的方法是可以改变返回值的类型。但是不能通过返回值进行 overload。

第十六，Set 里的元素是不能重复的，那么用什么方法来区分重复与否呢？ 是用==还是 equals()？ 它们有何区别？

Set 里的元素是不能重复的，那么用 iterator()方法来区分重复与否。equals()是判读两个 Set 是否相等。
equals()和==方法决定引用值是否指向同一对象 equals()在类中被覆盖，为的是当两个分离的对象的内容和类型相配的话，返回真值。

第十七，给我一个你最常见到的 runtime exception。（如果你这个答不出来，面试的人会认为你没有实际编程经验）

ArithmeticException, ArrayStoreException, BufferOverflowException, BufferUnderflowException, CannotRedoException, CannotUndoException, ClassCastException, CMMException, ConcurrentModificationException, DOMException, EmptyStackException, IllegalArgumentException, IllegalMonitorStateException, IllegalPathStateException, IllegalStateException, ImagingOpException, IndexOutOfBoundsException, MissingResourceException, NegativeArraySizeException, NoSuchElementException, NullPointerException, ProfileDataException, ProviderException, RasterFORMatException, SecurityException, SystemException, UndeclaredThrowableException, UnmodifiableSetException, UnsupportedOperationException

第十八，error 和 exception 有什么区别？

error 表示恢复不是不可能但很困难的情况下的一种严重问题。比如说内存溢出。不可能指望程序能处理这样的情况。
exception 表示一种设计或实现问题。也就是说，它表示如果程序运行正常，从不会发生的情况。
error--unchecked exception--checked

第十九，List，Set，Map 是否继承自 Collection 接口？

List，Set 是 Map 不是(与 Collection 同级)

第二十，abstract class 和 interface 有什么区别？ （常问）

声明方法的存在而不去实现它的类被叫做抽象类（abstract class），它用于要创建一个体现某些基本行为的类，并为该类声明方法，但不能在该类中实现该类的情况。不能创建 abstract 类的实例。然而可以创建一个变量，其类型是一个抽象类，并让它指向具体子类的一个实例。不能有抽象构造函数或抽象静态方法。Abstract 类的子类为它们父类中的所有抽象方法提供实现，否则它们也是抽象类为。取而代之，在子类中实现该方法。知道其行为的其它类可以在类中实现这些方法。
接口（interface）是抽象类的变体。在接口中，所有方法都是抽象的。多继承性可通过实现这样的接口而获得。接口中的所有方法都是抽象的，没有一个有程序体。接口只可以定义 static final 成员变量。接口的实现与子类相似，除了该实现类不能从接口定义中继承行为。当类实现特殊接口时，它定义（即将程序体给予）所有这种接口的方法。然后，它可以在实现了该接口的类的任何对象上调用接口的方法。由于有抽象类，它允许使用接口名作为引用变量的类型。通常的动态联编将生效。引用可以转换到接口类型或从接口类型转换，instanceof 运算符可以用来决定某对象的类是否实现了接口。

第二十一，abstract 的 method 是否可同时是 static,是否可同时是 native，是否可同时是 synchronized？

都不能

第二十二，接口是否可继承接口？ 抽象类是否可实现(implements)接口？ 抽象类是否可继承实体类(concrete class)？

接口可以继承接口。抽象类可以实现(implements)接口，抽象类是否可继承实体类，但前提是实体类必须有明确的构造函数。

第二十三，启动一个线程是用 run()还是 start()？

启动一个线程是调用 start()方法，使线程所代表的虚拟处理机处于可运行状态，这意味着它可以由 JVM 调度并执行。这并不意味着线程就会立即运行。run()方法可以产生必须退出的标志来停止一个线程。

第二十四，构造器 Constructor 是否可被 override？

构造器 Constructor 不能被继承，因此不能重写 Overriding，但可以被重载 Overloading。

第二十五，是否可以继承 String 类？

String 类是 final 类故不可以继承。

第二十六，当一个线程进入一个对象的一个 synchronized 方法后，其它线程是否可进入此对象的其它方法？

不能，一个对象的一个 synchronized 方法只能由一个线程访问。

第二十七，try {}里有一个 return 语句，那么紧跟在这个 try 后的 finally {}里的 code 会不会被执行，什么时候被执行，在 return 前还是后？

会执行，在 return 前执行。

第二十八，编程题：用最有效率的方法算出 2 乘以 8 等于几？（有 C 背景的程序员特别喜欢问这种问题）

2 << 3

第二十九，两个对象值相同(x.equals(y) == true)，但却可有不同的 hash code，这句话对不对？

不对，有相同的 hash code。看你的对象类型了。

第三十，当一个对象被当作参数传递到一个方法后,此方法可改变这个对象的属性,并可返回变化后的结果，那么这里到底是值传递还是引用传递？

是值传递。Java 编程语言只由值传递参数。当一个对象实例作为一个

参数被传递到方法中时，参数的值就是对该对象的引用。对象的内容可以在被调用的方法中改变，但对象的引用是永远不会改变的。

第三十一，swtich 是否能作用在 byte 上，是否能作用在 long 上，是否能作用在 String 上？

switch（expr1）中，expr1 是一个整数表达式。因此传递给 switch 和 case 语句的参数应该是 int、 short、 char 或者 byte。long,string 都不能作用于 swtich。

第三十二，编程题：写一个 Singleton 出来。

Singleton 模式主要作用是保证在 Java 应用程序中，一个类 Class 只有一个实例存在。
一般 Singleton 模式通常有几种种形式:
第一种形式：定义一个类，它的构造函数为 private 的，它有一个 static 的 private 的该类变量，在类初始化时实例话，通过一个 public 的 getInstance 方法获取对它的引用,继而调用其中的方法。

```
public class Singleton {
    private Singleton(){}
    //在自己内部定义自己一个实例，是不是很奇怪？
    //注意这是 private 只供内部调用
    private static Singleton instance = new Singleton();
    //这里提供了一个供外部访问本 class 的静态方法，可以直接访问

    public static Singleton getInstance() {
        return instance;
     }
}
```

第二种形式:

```
public class Singleton {
    private static Singleton instance = null;
    public static synchronized Singleton getInstance() {
    //这个方法比上面有所改进，不用每次都进行生成对象，只是第一次
    //使用时生成实例，提高了效率！
    if (instance==null)
        instance=new Singleton();
return instance;       }
}
```

其他形式:
定义一个类，它的构造函数为 private 的，所有方法为 static 的。
一般认为第一种形式要更加安全些