# 1第一阶段面试题

## 简述static和final的用法？

static：修饰属性，方法，代码块

（1）静态属性：也可叫类变量 类名.属性名 来访问

（共有的类变量与对象无关，只和类有关）

注意：类中的实例变量是在创建对象时被初始化的，被static修饰的属性，也就是类变量，是在类加载时被创建并进行初始化，类加载的过程是进行一次。也就是类变量只会被创建一次。

（2）静态方法：类名.方法名 直接访问

注意：static修饰的方法，不能直接访问本类中的非静态(static)成员(包括方法和属性)

本类的非静态方法可以访问本类的静态成员（包括方法和属性），可以调用静态方法。

修饰变量，方法，类

final：修饰变量，类，方法

（1）修饰变量

被fianl修饰的成员变量就是常量（常量名大写），一旦赋值不能改变

修饰局部变量：修饰基本数据类型 -> 变量的值不能改变

修饰引用 -> 引用只能指向固定的对象

修饰实例变量：默认值不生效，可以再赋值

（2）修饰方法 ：不能被子类覆盖

（3）修饰类：不能被继承

在一个final类中的所有方法，默认都是final的

注意：final，不能用来修饰构造方法。

## 写出冒泡排序的算法

for(int i=0;i<arr.length-1;i++){ //控制轮数

for(int j=0;j<arr.length-1-i;j++){ //控制每一轮的次数

if(arr[j]>arr[j+1]){ //每次都是和它下一个元素比

int t=arr[j];

arr[j]=arr[j+1];

arr[j+1]=t;

}

}

}

## abstract class 和 interface 有什么区别?

|  |  |  |
| --- | --- | --- |
|  | abstract class | interface |
| 实例化 | 不能 | 不能 |
| 类 | 一种继承关系，一个类只能使用一次继承关系。可以通过实现多个接口 | 一个类可以实现多个interface |
| 数据成员 | 可有自己的 | 静态的不能被修改即必须是static final，一般不在此定义 |
| 方法 | 可以私有的，非abstract方法，必须实现 | 不能有私有的，默认是public，abstract 类型 |
| 变量 | 可有私有的，变量有默认的类型，其值可以在子类中重新定义，也可以重新赋值 | 不可有私有的，默认是public static final 型，且必须给其初值，实现类中不能重新定义，不能改变其值。 |
| 设计理念 | 表示的是“is-a”关系 | 表示的是“like-a”关系 |
| 实现 | 需要继承，要用extends | 要用implements |

1.相同点

A. 两者都是抽象类，都不能实例化。

B. interface实现类及abstrct class的子类都必须要实现已经声明的抽象方法。

2. 不同点

A. interface需要实现，要用implements，而abstract class需要继承，要用extends。

B. 一个类可以实现多个interface，但一个类只能继承一个abstract class。

C. interface强调特定功能的实现，而abstract class强调所属关系。

D. 尽管interface实现类及abstrct class的子类都必须要实现相应的抽象方法，但实现的形式不同。interface中的每一个方法都是抽象方法，都只是声明的 (declaration, 没有方法体)，实现类必须要实现。而abstract class的子类可以有选择地实现。

## 类有哪三个基本特性？各特性的优点？

类具有封装性、继承性和多态性。

封装性：类的封装性为类的成员提供公有、缺省、保护和私有等访问权限，目的是隐藏类中的私有变量和类中方法的实现细节。

继承性：允许通过继承原有类的某些特性或全部特性而产生全新的类，原有的累称为父类，产生的新类称为子类。子类不仅可以直接继承父类的共性，而且也可以创建它特有的个性。

多态性：是指在基类中定义的属性和方法被子类继承之后，可以具有不同的数据类型或表现出不同行为，多态性有两种表现形式：重载和覆盖。

## Error和Exception有什么区别? 列出你见过的Exception并简要说明

答：error表示系统级的错误和程序不必处理的异常，

是恢复不是不可能但很困难的情况下的一种严重问题；

比如内存溢出，不可能指望程序能处理这样的情况；

exception表示需要捕捉或者需要程序进行处理的异常，

是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。 常见异常有： NullPointerException：

当操作一个空引用时会出现此错误。 NumberFormatException：

数据格式转换出现问题时出现此异常。 ClassCastException：

强制类型转换类型不匹配时出现此异常。 ArrayIndexOutOfBoundsException：

数组下标越界，当使用一个不存在的数组下标时出现此异常。

## java中会存在内存泄露吗？请简单描述。

答：内存泄露是指系统中存在无法回收的内存，

有时候会造成内存不足或系统崩溃。Java存在内存泄露。

Java中的内存泄露当然是指：存在无用但是垃圾回收器无法回收的对象。

而且即使有内存泄露问题存在，也不一定会表现出来。

自己实现堆栈的数据结构时有可能会出现内存泄露。

## 多线程有几种实现方法,都是什么?同步有几种实现方法,都是什么?

答：多线程有两种实现方法：继承Thread类或者实现Runnable接口。

实现同步也有两种方法：一种是同步方法，另一种是同步代码块。

同步方法是在方法返回类型前面加上synchronized关键字

同步代码块是synchronized (这里写需要同步的对象){...}

## .sleep()和wait()有什么区别？

答案：

（2）不同点：

1. Thread类的方法：sleep(),yield()等

Object的方法：wait()和notify()等

2.每个对象都有一个锁来控制同步访问。Synchronized关键字可以和对象的锁交互，来实现线程的同步。

sleep方法没有释放锁，

wait方法释放了锁，使得其他线程可以使用同步控制块或者方法。

3.wait，notify和notifyAll只能在同步控制方法或者同步控制块里面使用，而sleep可以在任何地方使用

4.sleep必须捕获异常，而wait，notify和notifyAll不需要捕获异常

## 谈谈java跟你所知道的其它的语言相比，有什么优点？？

Java是一种可以撰写跨平台应用程序的面向对象的[程序设计语言](https://www.baidu.com/s?wd=%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1%E8%AF%AD%E8%A8%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9uAFBuWb4PH--mh7Wmhc40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHT4P1Rkn10L)。Java技术具有卓越的通用性、高效性、平台移植性和安全性，广泛应用于PC、数据中心、游戏控制台、科学超级计算机、移动电话和互联网，同时拥有全球最大的开发者专业社群。  
Java是功能完善的通用[程序设计语言](https://www.baidu.com/s?wd=%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1%E8%AF%AD%E8%A8%80&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9uAFBuWb4PH--mh7Wmhc40ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHT4P1Rkn10L)，可以用来开发可靠的、要求严格的应用程序。  
java是纯面向对象开发，功能强大，分支众多，没有java不能做的软件。C/S也好B/S也好。从功能上讲，没有语言可以和java相比。

C是面向过程编程的，这样往往会导致所谓的单一程序，既所有的功能只能包含在几个（通常是一个）代码模块中。当然，C语言也有自身的不足，比如：C语言的语法限制不太严格，对变量的类型约束不严格，影响程序的安全性，对数族下标越界不作检查等。从应用的角度，C语言比其他高级语言较难掌握。

## 谈谈你对面向对象的理解？？

所谓的面向对象就是将我们的程序模块化，对象化，把具体事物的特性属性和通过这些属性来实现一些动作的具体方法放到一个类里面，这就是封装。封装是我们所说的面相对象编程的特征之一。除此之外还有继承和多态。继承有点类似与我们生物学上的遗传，就是子类的一些特征是来源于父类的，儿子遗传了父亲或母亲的一些性格，或者相貌，又或者是运动天赋。有点种瓜得瓜种豆得豆的意思。面向对象里的继承也就是父类的相关的属性，可以被子类重复使用，子类不必再在自己的类里面重新定义一回，父类里有点我们只要拿过来用就好了。而对于自己类里面需要用到的新的属性和方法，子类就可以自己来扩展了。当然，会出现一些特殊情况，就是我们在有一些方法在父类已经定义好了，但是子类我们自己再用的时候，发现，其实，我们的虽然都是计算工资的，但是普通员工的工资计算方法跟经理的计算方法是不一样的，所以这个时候，我们就不能直接调用父类的这个计算工资的方法了。这个时候我们就需要用到面向对象的另一个特性，多态。对，就是多态，我们要在子类里面把父类里面定义计算工资的方法在子类里面重新实现一遍。多态包含了重载和重写。重写很简单就是把子类从父亲类里继承下来的方法重新写一遍，这样，父类里相同的方法就被覆盖了，当然啦，你还是可以通过super.CaculSalary方法来调用父类的工资计算方法。而重载就是类里面相同方法名，不同形参的情况，可以是形参类型不同或者形参个数不同，或者形参顺序不同，但是不能使返回值类型不同。

## 简单讲一下java的跨平台原理

java源程序（.java文件）通过编译器编译成为Class文件（字节码文件），而它的class文件是基于字节码（以byte为单位存储的文件）的，而字节码文件是描述程序要运行的的虚指令的集合，这些虚指令的集合与任何的平台无关，Java虚拟机认识它（只要在不同的平台下部署相应的jre，运行jvm！就可以了）

## 有了基本数据类型，为什么还需要包装类型？

我们知道Java是一个面相对象的编程语言，基本类型并不具有对象的性质，为了让基本类型也具有对象的特征，就出现了包装类型（如我们在使用集合类型Collection时就一定要使用包装类型而非基本类型），它相当于将基本类型“包装起来”，使得它具有了对象的性质，并且为其添加了属性和方法，丰富了基本类型的操作。

另外，当需要往ArrayList，HashMap中放东西时，像int，double这种基本类型是放不进去的，因为容器都是装object的，这是就需要这些基本类型的包装器类了。

## 说一下"=="和equals方法究竟有什么区别？

==操作符专门用来比较两个变量的值是否相等，也就是用于比较变量所对应的内存中所存储的数值是否相同，要比较两个基本类型的数据或两个引用变量是否相等，只能用==操作符。

equals 方法是用于比较两个独立对象的内容是否相同，就好比去比较两个人的长相是否相同，它比较的两个对象是独立的。

## 讲一下java中的集合

set(集）、list(列表）和map(映射)。   
区别嘛 HASHMAP只有KEY和value值对应的。。set是可以自动清楚相同的元素  
list是其对象以线性方式存储，没有特定顺序，只有一个开头和一个结尾，当然，它与根本没有顺序的集是不同的。   
列表在数据结构中分别表现为：数组和向量、链表、堆栈、队列。

## ArrayList和LinkedList的区别?

1.ArrayList是实现了基于动态数组的数据结构，LinkedList基于链表的数据结构。   
2.对于随机访问get和set，ArrayList觉得优于LinkedList，因为LinkedList要移动指针。   
3.对于新增和删除操作add和remove，LinedList比较占优势，因为ArrayList要移动数据。

## 讲一下HashMap和HashTable的区别?

在Java 2以前，一般使用Hashtable来映射键值和元素。为了使用Java集合框架，Java对Hashtable进行了重新设计，但是，为了向后兼容保留了所有的方法。Hashtable实现了Map接口，除了Hashtable具有同步功能之外，它与HashMap的用法是一样的。·  
在使用时一般是用ArrayList代替Vector，LinkedList代替Stack，HashMap代替HashTable，即使在多线程中需要同步，也是用同步包装类。

## 请说出集合类中List、Map、Set的区别

答：List和Set继承了Collection接口，而map不是；

List中存放元素有顺序并且可以重复；

set中存放的元素是无序并且是不可能重复的；

Map中存放是键值对。

## Collection 和 Collections的区别？

答：Collection是java.util下的接口，它是各种集合的父接口，

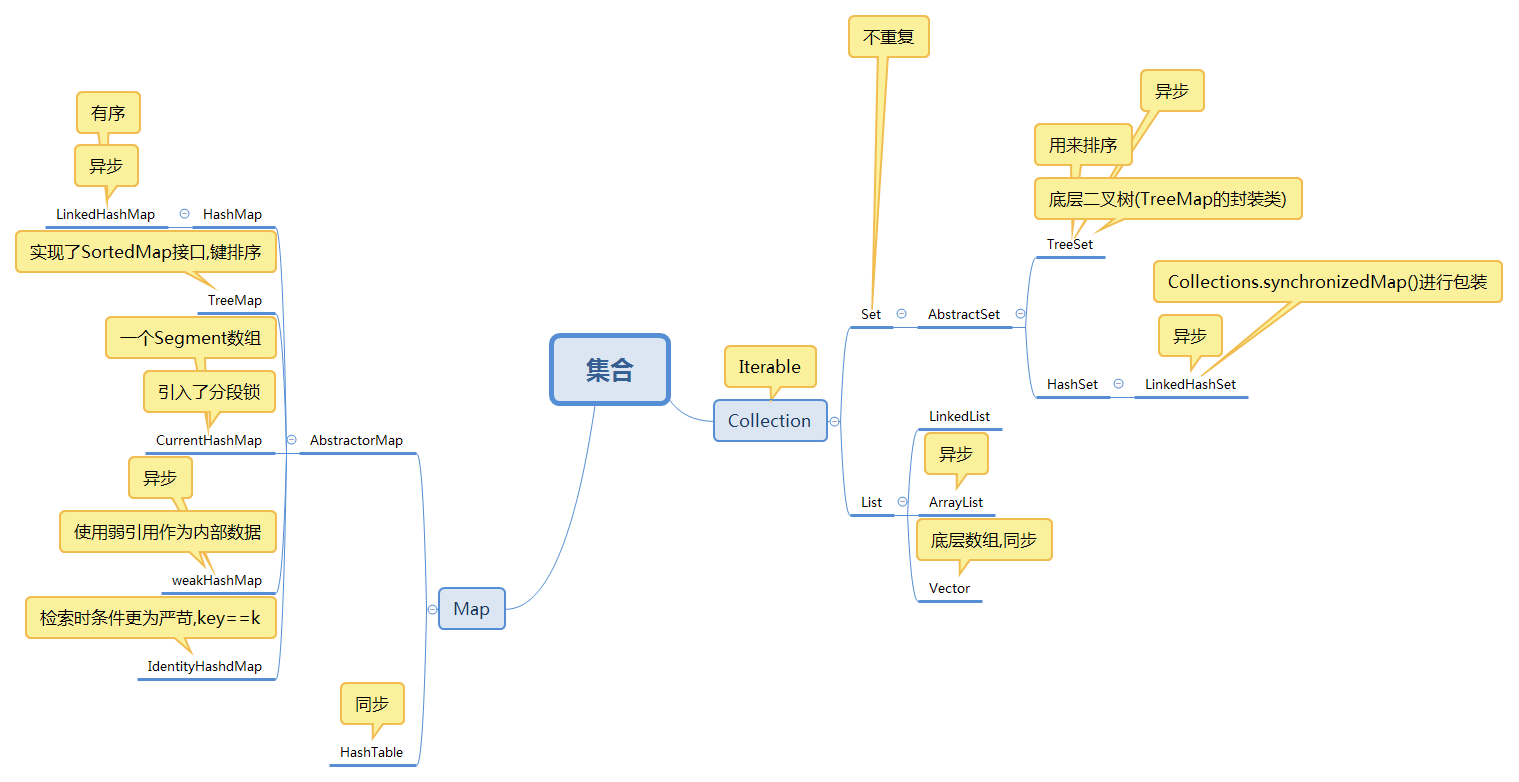
继承于它的接口主要有Set 和List；Collections是个java.util下的类，

是针对集合的帮助类，提供一系列静态方法实现对各种集合的搜索、排序、线程安全化等操作。

HashSet :内部封装了HashMap和TreeMap,数据保存在键这一列

LinkedHashMap(哈希表,存放数据以链表来连接,数组存放无序但链表连接有序,可看做一个有序的HashMap),既可快速定位,查找数据,又可以使数据存放有序

ConcurrentHashMap(分段加锁)Concurrent-同步



## String、StringBuffer和StringBuilder的区别?

1. 首先说运行速度，或者说是执行速度，在这方面运行速度快慢为：StringBuilder > StringBuffer > String

2. 再来说线程安全

在线程安全上，StringBuilder是线程不安全的，而StringBuffer是线程安全的

String：适用于少量的字符串操作的情况

　　StringBuilder：适用于单线程下在字符缓冲区进行大量操作的情况

StringBuffer：适用多线程下在字符缓冲区进行大量操作的情况

## 讲一下线程的几种实现方式?

extends Thread   
implements Runnable   
implements Callable

Runnable和Callable的区别是，   
（1）Callable规定的方法是call（），Runnable规定的方法是run（）。   
（2）Callable的任务执行后可返回值，而Runnable的任务是不能返回值得   
（3）call方法可以抛出异常，run方法不可以   
（4）运行Callable任务可以拿到一个Future对象，表示异步计算的结果。

## 讲一下线程的几种启动方式？

.第一种方法是将类声明为 Thread 的子类。该子类应重写 Thread 类的 run 方法，然后在run方法里填写相应的逻辑代码。

第二种方法是实现Runnable接口，并编写run方法,相比继承Thread类创建线程的好处是以实现接口的方式创建线程可以对类进行更好的扩展，该类可以继承其他类来扩展自身需求，相比第一种方式更加灵活，扩展性强。

实现Callable接口创建线程与Runnable接口的不同之处在于：如果你想要在线程执行完毕之后得到带有返回值的线程则实现Callable接口

## 有没有使用过线程并发库?

在java5之后，就有了线程池的功能了，在介绍线程池之前，先来简单看一下线程池的概念。假设我开了家咨询公司，那么每天会有很多人过来咨询问题，如果我一个个接待的话，必然有很多人要排队，这样效率就很差，我想解决这个问题，现在我雇几个客服，来了一个咨询的，我就分配一个客服去接待他，再来一个，我再分配个客服去接待……如果第一个客服接待完了，我就让她接待下一个咨询者，这样我雇的这些客服可以循环利用。这些客服就好比不同的线程，那么装这些线程的容器就称为线程池。

## 静态变量和实例变量的区别？

静态变量也叫类变量，这种变量前加了static修饰符。可以直接用类名调用，也可以用对象调用，而且所有对象的同一个类变量 都是共享同一块内存空间。

       实例变量也叫对象变量，这种变量没有加static修饰符。只能通过对象调用， 而且所有对象的同一个实例变量是共享不同的内存空间的。

区别在于：

       静态变量是所有对象共有的，某一个对象将它的值改变了，其他对象再去获取它的值，得到的是改变后的值；

       实例变量则是每一个对象私有的，某一个对象将它的值改变了，不影响其他对象取值的结果，其他对象仍会得到实例变量一开始就被赋予的值。

## 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 try{}中的return执行后在没有返回数据时先去执行finally{}中的代码，然后再返回。所以说finally{}在return中间执行

## 同步和异步有何异同，在什么情况下分别使用他们？举例说明。

如果数据将在线程间共享.例如正在写的数据以后可能被另一个线程读到,或者正在读的数据可能已经被另一个线程写过了,那么这些数据就是共享数据,必须进行同步存取.

当应用程序在对象上调用了一个需要花费很长时间来执行的方法,并且不希望让程序等待方法的返回时,就应该使用异步编程,在很多情况下采用异步途径往往更有效率.

## java 中有几种类型的流？JDK 为每种类型的流提供了一些抽象类以供继承， 请说出他们分别是哪些类？

**Java中的流分为两种，一种是字节流，另一种是字符流，分别由四个抽象类来表示（每种流包括输入和输出两种所以一共四个）:InputStream，OutputStream，Reader，Writer。Java中其他多种多样变化的流均是由它们派生出来的.**

## &和&&的区别？

&和&&都可以用作逻辑运算符，表示逻辑与。当运算符两边的表达式都为true时，结果才为true；否则，结果为false。

另外&&还具有短路功能，也就是说，当&&左边的表达式结果为false时，将不再运算&&右边的表达式，结果肯定为false。例如，对于if(str!=null&&!str.equals(“”)),当str为null时，不会对&&右边的表达式进行运算，否则会出现空指针异常。

&还可以用作位运算符，当&两边的表达式不是boolean类型时，&表示按位与。

## 数组有没有length()这个方法? String有没有length()这个方法？

数组中没有length()这个方法，但是数组中有length这个属性。用来表示数组的长度。

      String中有length()这个方法。用来得到字符串的长度。

## 构造器Constructor是否可被override?

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

Constructor不能被继承，所以Constructor也就不能被override。每一个类必须有自己的[构造函数](https://www.baidu.com/s?wd=%E6%9E%84%E9%80%A0%E5%87%BD%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9ujI9nWbkn1bdujfsP1Ds0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bkrjmkPWDv)，负责构造自己这部分的构造。子类不会覆盖父类的[构造函数](https://www.baidu.com/s?wd=%E6%9E%84%E9%80%A0%E5%87%BD%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9ujI9nWbkn1bdujfsP1Ds0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bkrjmkPWDv)，相反必须负责在一开始调用父类的[构造函数](https://www.baidu.com/s?wd=%E6%9E%84%E9%80%A0%E5%87%BD%E6%95%B0&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9ujI9nWbkn1bdujfsP1Ds0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1bkrjmkPWDv)。

## **构造器如何工作？**

Java在构造实例时的顺序是这样的：

  1、分配对象空间，并将对象中成员初始化为0或者空，java不允许用户操纵一个不定值的对象。

2、执行属性值的显式初始化

3、执行构造器

4 、将变量关联到堆中的对象上

## super与this的区别？

不同点：   
1、super()主要是对父类构造函数的调用，this()是对重载构造函数的调用   
2、super()主要是在继承了父类的子类的构造函数中使用，是在不同类中的使用；this()主要是在同一类的不同构造函数中的使用   
相同点：   
1、super()和this()都必须在构造函数的第一行进行调用，否则就是错误的

## GC是什么? 为什么要有GC?

GC是垃圾收集的意思，内存处理是编程人员容易出现问题的地方，

忘记或者错误的内存回收会导致程序或系统的不稳定甚至崩溃，Java提供的GC功能可以自动监测对象是否超过作用域从而达到自动回收内存的目的，

Java语言没有提供释放已分配内存的显示操作方法。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()或Runtime.getRuntime().gc()，但JVM可以屏蔽掉显示的垃圾回收调用。

垃圾回收可以有效的防止内存泄露，有效的使用可以使用的内存。垃圾回收器通常是作为一个单独的低优先级的线程运行，不可预知的情况下对内存堆中已经死亡的或者长时间没有使用的对象进行清除和回收，程序员不能实时的调用垃圾回收器对某个对象或所有对象进行垃圾回收。

在Java诞生初期，垃圾回收是Java最大的亮点之一，因为服务器端的编程需要有效的防止内存泄露问题，然而时过境迁，如今Java的垃圾回收机制已经成为被诟病的东西。移动智能终端用户通常觉得iOS的系统比Android系统有更好的用户体验，其中一个深层次的原因就在于Android系统中垃圾回收的不可预知性。

## 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？

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

## 内部类可以引用他包含类的成员吗？有没有什么限制？

完全可以。如果不是静态内部类，那没有什么限制！  
 如果你把静态嵌套类当作内部类的一种特例，那在这种情况下不可以访问外部类的普通成员变量，而只能访问外部类中的[静态成员](https://www.baidu.com/s?wd=%E9%9D%99%E6%80%81%E6%88%90%E5%91%98&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1d9uhD3mWNhPW-9nWTvPHbL0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EPHDvrjnLnHmz)

## Java有没有goto？ 答：goto 是Java中的保留字，在目前版本的Java中没有使用。（根据James Gosling（Java之父）编写的《The Java Programming Language》一书的附录中给出了一个Java关键字列表，其中有goto和const，但是这两个是目前无法使用的关键字，因此有些地方将其称之为保留字，其实保留字这个词应该有更广泛的意义，因为熟悉C语言的程序员都知道，在系统类库中使用过的有特殊意义的单词或单词的组合都被视为保留字）

## **解释内存中的栈(stack)、堆(heap)和静态区(static area)的用法**

答：通常我们定义一个基本数据类型的变量，一个对象的引用，还有就是函数调用的现场保存都使用内存中的栈空间；而通过new关键字和构造器创建的对象放在堆空间；程序中的字面量（literal）如直接书写的100、”hello”和常量都是放在静态区中。栈空间操作起来最快但是栈很小，通常大量的对象都是放在堆空间，理论上整个内存没有被其他进程使用的空间甚至硬盘上的虚拟内存都可以被当成堆空间来使用。

## **用最有效率的方法计算2乘以8？**

答： 2 << 3（左移3位相当于乘以2的3次方，右移3位相当于除以2的3次方）。

## **在Java中，如何跳出当前的多重嵌套循环？**

答：在最外层循环前加一个标记如A，然后用break A;可以跳出多重循环。（Java中支持带标签的break和continue语句，作用有点类似于C和C++中的goto语句，但是就像要避免使用goto一样，应该避免使用带标签的break和continue，因为它不会让你的程序变得更优雅，很多时候甚至有相反的作用，所以这种语法其实不知道更好）

## 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？

答：不对，如果两个对象x和y满足x.equals(y) == true，它们的哈希码（hash code）应当相同。Java对于eqauls方法和hashCode方法是这样规定的：(1)如果两个对象相同（equals方法返回true），那么它们的hashCode值一定要相同；(2)如果两个对象的hashCode相同，它们并不一定相同。当然，你未必要按照要求去做，但是如果你违背了上述原则就会发现在使用容器时，相同的对象可以出现在Set集合中，同时增加新元素的效率会大大下降（对于使用哈希存储的系统，如果哈希码频繁的冲突将会造成存取性能急剧下降）。

## **char 型变量中能不能存贮一个中文汉字，为什么？**

答：char类型可以存储一个中文汉字，因为Java中使用的编码是Unicode（不选择任何特定的编码，直接使用字符在字符集中的编号，这是统一的唯一方法），一个char类型占2个字节（16比特），所以放一个中文是没问题的。

## Anonymous Inner Class(匿名内部类)是否可以继承其它类？是否可以实现接口？

答：可以继承其他类或实现其他接口，在Swing编程和Android开发中常用此方式来实现事件监听和回调。

## 内部类可以引用它的包含类（外部类）的成员吗？有没有什么限制？

答：一个内部类对象可以访问创建它的外部类对象的成员，包括私有成员。

## Java 中的final关键字有哪些用法？

答：(1)修饰类：表示该类不能被继承；(2)修饰方法：表示方法不能被重写；(3)修饰变量：表示变量只能一次赋值以后值不能被修改（常量）。

## 写出常见的5个RuntimeException

(1)java.lang.NullPointerException空指针异常，出现原因：调用了未经初始化的对性爱那个或者不存在的对象。

(2)ClassNoFoundException 指定类找不到，出现原因：类的名称和路径加载错误，通常是试图通过字符串来加载某个类时可能引发异常。

(3)NumberFormatException字符串转换为数字异常，出现原因：字符串数据中包含非数字型字符。

(4)IndexOutOfBoundsException数组下标越界异常

(5)IllegalArgumentException 方法传递参数错误

(6)ClassCastException数据类型转换异常

(7)NoClassDefFoundExcetion 未找到类定义错误

(8)SQLException SQL异常

(9)InstantiationException实例化异常

(10)NoSuchMethodExceptioin 方法不存在异常

## 阐述final、finally、finalize的区别。

答：  
- final：修饰符（关键字）有三种用法：如果一个类被声明为final，意味着它不能再派生出新的子类，即不能被继承，因此它和abstract是反义词。将变量声明为final，可以保证它们在使用中不被改变，被声明为final的变量必须在声明时给定初值，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，不能在子类中被重写。  
- finally：通常放在try…catch…的后面构造总是执行代码块，这就意味着程序无论正常执行还是发生异常，这里的代码只要JVM不关闭都能执行，可以将释放外部资源的代码写在finally块中。  
- finalize：Object类中定义的方法，Java中允许使用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销毁对象时调用的，通过重写finalize()方法可以整理系统资源或者执行其他清理工作。

## **阐述ArrayList、Vector、LinkedList的存储性能和特性。**

答：ArrayList 和Vector都是使用数组方式存储数据，此数组元素数大于实际存储的数据以便增加和插入元素，它们都允许直接按序号索引元素，但是插入元素要涉及数组元素移动等内存操作，所以索引数据快而插入数据慢，Vector中的方法由于添加了synchronized修饰，因此Vector是线程安全的容器，但性能上较ArrayList差，因此已经是Java中的遗留容器。LinkedList使用双向链表实现存储（将内存中零散的内存单元通过附加的引用关联起来，形成一个可以按序号索引的线性结构，这种链式存储方式与数组的连续存储方式相比，内存的利用率更高），按序号索引数据需要进行前向或后向遍历，但是插入数据时只需要记录本项的前后项即可，所以插入速度较快。Vector属于遗留容器（Java早期的版本中提供的容器，除此之外，Hashtable、Dictionary、BitSet、Stack、Properties都是遗留容器），已经不推荐使用，但是由于ArrayList和LinkedListed都是非线程安全的，如果遇到多个线程操作同一个容器的场景，则可以通过工具类Collections中的synchronizedList方法将其转换成线程安全的容器后再使用（这是对装潢模式的应用，将已有对象传入另一个类的构造器中创建新的对象来增强实现）。

## **Collection和Collections的区别？**

答：Collection是一个接口，它是Set、List等容器的父接口；Collections是个一个工具类，提供了一系列的静态方法来辅助容器操作，这些方法包括对容器的搜索、排序、线程安全化等等。

## **List、Map、Set三个接口存取元素时，各有什么特点？**

答：List以特定索引来存取元素，可以有重复元素。Set不能存放重复元素（用对象的equals()方法来区分元素是否重复）。Map保存键值对（key-value pair）映射，映射关系可以是一对一或多对一。Set和Map容器都有基于哈希存储和排序树的两种实现版本，基于哈希存储的版本理论存取时间复杂度为O(1)，而基于排序树版本的实现在插入或删除元素时会按照元素或元素的键（key）构成排序树从而达到排序和去重的效果。

## 当一个线程进入一个对象的synchronized方法A之后，其它线程是否可进入此对象的synchronized方法B？

答：不能。其它线程只能访问该对象的非同步方法，同步方法则不能进入。因为非静态方法上的synchronized修饰符要求执行方法时要获得对象的锁，如果已经进入A方法说明对象锁已经被取走，那么试图进入B方法的线程就只能在等锁池（**注意不是等待池哦**）中等待对象的锁。

## 请说出与线程同步以及线程调度相关的方法。

答：  
- wait()：使一个线程处于等待（阻塞）状态，并且释放所持有的对象的锁；  
- sleep()：使一个正在运行的线程处于睡眠状态，是一个静态方法，调用此方法要处理InterruptedException异常；  
- notify()：唤醒一个处于等待状态的线程，当然在调用此方法的时候，并不能确切的唤醒某一个等待状态的线程，而是由JVM确定唤醒哪个线程，而且与优先级无关；  
- notityAll()：唤醒所有处于等待状态的线程，该方法并不是将对象的锁给所有线程，而是让它们竞争，只有获得锁的线程才能进入就绪状态；

## **Java中如何实现序列化，有什么意义？**

答：序列化就是一种用来处理对象流的机制，所谓对象流也就是将对象的内容进行流化。可以对流化后的对象进行读写操作，也可将流化后的对象传输于网络之间。序列化是为了解决对象流读写操作时可能引发的问题（如果不进行序列化可能会存在数据乱序的问题）。  
要实现序列化，需要让一个类实现Serializable接口，该接口是一个标识性接口，标注该类对象是可被序列化的，然后使用一个输出流来构造一个对象输出流并通过writeObject(Object)方法就可以将实现对象写出（即保存其状态）；如果需要反序列化则可以用一个输入流建立对象输入流，然后通过readObject方法从流中读取对象。序列化除了能够实现对象的持久化之外，还能够用于对象的深度克隆.

## **如何通过反射创建对象？**

答：  
- 方法1：通过类对象调用newInstance()方法，例如：String.class.newInstance()  
- 方法2：通过类对象的getConstructor()或getDeclaredConstructor()方法获得构造器（Constructor）对象并调用其newInstance()方法创建对象，例如：String.class.getConstructor(String.class).newInstance(“Hello”);

## 谈谈 JVM 的内存结构和内存分配？

Java 内存模型

Java 虚拟机将其管辖的内存大致分三个逻辑部分：方法区(Method Area)、 Java 栈和 Java 堆。

1、方法区是静态分配的，编译器将变量绑定在某个存储位置上，而且这些绑定不会在运行时改变。常数池，源代码中的命名常量、 String 常量和 static 变量保存在方法区。

2、 Java Stack 是一个逻辑概念，特点是后进先出。一个栈的空间可能是连续的，也可能是不连续的。最典型的 Stack 应用是方法的调用， Java 虚拟机每调用一次方法就创建一个方法帧（frame），退出该

方法则对应的 方法帧被弹出(pop)。栈中存储的数据也是运行时确定的。

3、 Java 堆分配(heap allocation)意味着以随意的顺序，在运行时进行存储空间分配和收回的内存管理模型。堆中存储的数据常常是大小、数量和生命期在编译时无法确定的。 Java 对象的内存总是在 heap 中分配。

b) java 内存分配

1、基础数据类型直接在栈空间分配;

2、方法的形式参数，直接在栈空间分配，当方法调用完成后从栈空间回收;

3、引用数据类型，需要用 new 来创建，既在栈空间分配一个地址空间，又在堆空间分配对象的类变量;

4、方法的引用参数，在栈空间分配一个地址空间，并指向堆空间的对象区，当方法调用完后从栈空间回收;

5、局部变量 new 出来时，在栈空间和堆空间中分配空间，当局部变量生命周期结束后，栈空间立刻被回收，堆空间区域等待 GC 回收;

6、方法调用时传入的实际参数，先在栈空间分配，在方法调用完成后从栈空间释放;

7、字符串常量在 DATA 区域分配 ， this 在堆空间分配;

8、数组既在栈空间分配数组名称， 又在堆空间分配数组实际的大小。

## 什么是双亲委派机制

双亲委派模式的工作原理的是;如果一个类加载器收到了类加载请求，它并不会自己先去加载，而是把这个请求委托给父类的加载器去执行，如果父类加载器还存在其父类加载器，则进一步向上委托，依次递归，请求最终将到达顶层的启动类加载器，如果父类加载器可以完成类加载任务，就成功返回，倘若父类加载器无法完成此加载任务，子加载器才会尝试自己去加载，这就是双亲委派模式，即每个儿子都不愿意干活，每次有活就丢给父亲去干，直到父亲说这件事我也干不了时，儿子自己想办法去完成，这不就是传说中的双亲委派模式.那么这种模式有什么作用呢?

## Static 关键字有哪些作用?

Static 有两种作用： 第一， 为某特定数据类型或对象分配单一的存储空间，而与创建对象的个数无关。 第二， 实现某个方法或属性与类而不是对象关联在一起。 Static 主要是有 4 种使用情况： Static 修饰成员变量：用 static 修饰的变量称之为静态变量或者也叫类变量/全局变量。静态变量 是随着类的加载而加载到方法区中的静态区，并且在静态区中赋予了初始值。静态变量是在对 象之前产生，所以可以不通过对象来调用，而是通过类来调用，所以可以通过类名.静态变量的 方式调用静态变量。由于每个对象在堆内存中存储的是静态变量在静态区中的地址，所以所有 的对象本质上共用一个静态变量。 Static 修饰成员方法：用static 修饰的方法就称之为静态方法，也叫类方法。静态方法在类加载 的时候加载到了方法区中的静态区，在调用的时候到栈内存中执行。---静态方法是先于对象而 存在的。静态方法可以通过类名.方法名的方式来调用执行。 静态代码块：静态代码块在类中是独立于成员变量和成员方法的代码块。它不在任何一个方法 体内，jvm 在加载类时会执行静态代码块，静态代码块是先于构造代码块执行。静态代码块在 类加载的时候执行，只执行一次。 静态内部类：静态内部类是指被声明为静态内部类，它可以不依赖与外部类实例对象而被实例 化，而通常的内部类需要在外部类实例化后才能实例化。静态内部类不能与外部类有相同的名 字，不能访问外部类的普通成员变量，只能访问外部类中的静态成员和静态方法

## Instanceof 有什么作用？

Instanceof 的作用是判断一个引用类型的变量所指向的对象是否是一个类的实例。

## 什么是不可变类？

不可变类是指当创建了这个类的实例后，就不允许修改它的值了，也就是说，一个对象一旦被 创建出来，在其整个生命周期中，它的成员变量就不能被修改了。

## 值传递与引用传递的区别？

值传递：在方法调用中，实参会把它的值传递给形参，形参只是用实参的值初始化一个临时的 存储单元（方法内的局部变量），因此性参与实参虽然有着相同的值，但是却有着不用的存储单 元，因此对形参的改变不会影响实参的值。 引用传递：在方法的调用中，传递的是对象（也可以看做是对象的地址），这时形参与实参指向 同一块存储单元（对象），因此对形参的修改就会影响实参的值。

## 强制类型转换的注意事项有哪些？

Java 在涉及 byte、short 和 char 类型的运算时，首先会把这些类型的变量值强制转换为 int 类型， 然后对 int 类型的值进行计算，最后得到的值也是 int 类型。因此，如果把两个 short 类型的值相 加，最后得到的结果是 int 型；如果把两个 byte 类型的值相加，最后也会得到一个 int 类型的值。 如果需要得到 short 类型的结果，就必须显示地把运算结果转换为 short 类型。

## ++i与 i++的区别？

++i 先运算后赋值，i++先赋值后运算。

## 字符串存储的机制是什么？

字符串会存储在常量池中。在给字符串赋值的时候，JVM 会检查常量池中是否已经存在该字符 串，如果存在则直接引用该地址，否则会在常量池中创建该字符串然后引用该地址