各位同学，面试是我们就业**最关键也是最简单**的一步，如果面试都过不了何来就业？简单是因为它是通过**平常的积累就可以获得这项技能**，所以我们必须对它足够重视。综合实际情况，整理出面试中经常问到JavaSE面试题，这些题目有些需要大家**去网上查资料然后用自己的思路总结，千万不要标准化，一定要有自己的想法。在查询这些资料的同时也会加深你对编程的理解，大家务必自己动手不要去借鉴其他同学的**。

此阶段周期时间为20天，中午面试时间二十分钟，三个人为一组，每个人时间为4分钟。

提前两天将要面试的题目给出来，由班长来主持，随机抽取三人，面试完之后由技术班长进行点评，然后再随机抽取一位同学进行复述当天的面试题。

此阶段要达到的效果：

经过这个阶段，学员的演讲能力都会得到明显的提升，经过准备之后都能演讲8-10分钟，并且可以开始有一定的扩展能力。

## JavaSE面试题

1. 抽象类和接口的区别

抽象类和接口

1. 接口里面只能存在public abstract 方法，不能够对方法进行实现，里面的方法都是抽象的方法。，而抽象类可以选择性的提供成员方法的实现细节。
2. 接口中的成员变量只能是public static final类型的。，抽象类中的成员变量可以是各种类型的，子类中可以修改
3. 接口中不能含有静态代码块，以及静态的方法。而抽象类可以有静态代码块和静态方法；

4，继承接口时，必须对接口内部的所有方法进行实现，并且可以进行多重的接口继承，继承抽象类时，可以选择的进行实现方法，并且只能进行单继承。

5，多态的实现时，如果继承了一个抽象类，那么可以使用父类的对象来接收子类。

如果继承了多个接口，可以使用多个接口类去接收子类。继承抽象类的属性是子类和父类，继承接口，属性就非常多了。  
6，抽象类是对事物对象的整体抽象，比如动物的抽象父类，cat dog猫继承动物的抽象类。而接口是对一些功能行为的抽象，如动物必须具有的一些功能。

7，抽象类中可以有构造器，接口中不可以有。

1. 重载和重写的区别
2. 重载是发生在同一个类里面的，方法名必须相同，参数类型不同，参数个数不同，参数顺序不同，方法返回值不同，
3. 重写是发生在子类和父类之间的，方法名和参数列表必须相同，返回值范围小于等于父类，抛出的异常范围小于等于父类，访问修饰符小于等于父类、
4. stringBuffer和stringBuilder的区别

AbstractStringBuilder是StringBuilder与StringBuffer的公共父类

1. 线程安全性：

StringBuffer对方法加了同步锁或者对调用的方法加了同步锁，所以是线程安全的。StringBuilder并没有对方法进行加同步锁，所以是非线程安全的。

1. 性能

StringBuilder效率高，

1. **集合的体系架构（单列集合和双列集合）**

集合处于java.util，

Collection的接口，

Map接口，Set接口，List接口、

Arrlist集合继承List接口，

LinkList集合继承List接口

Vector集合继承List接口

HashSet TreeSet继承自Set接口

TreeMap,HashMap,Properties,HashTable继承Map接口

1. **ArrayList和LinkedList的区别**

**ArraryList是动态数组，LinkedList是链表的实现。**

1. **动态数组在内存的存储是一串连续的内存空间。在查询、删除、插入、增加的时候可以使用索引进行相应的操作，同时在插入和删除的时候，必须在索引处向后移动。**
2. **链表在内存中不是连续的内存空间，在查询、删除、插入、增加的时候，只能从头开始遍历，到达索引对应的位置的时候，才能进行相应的操作。在插入、删除的时候，不需要向后移动，直接在索引处的前后，对节点的前后指针进行操作。**
3. **hashMap和HashTable的区别**
4. java的GC机制的理解

GC机制的目的在于，不定时的清除不再使用的对象，回收对应的被其使用的内存空间的机制。

GC通过确定对象是否被活动的对象引用来确定是否收回收该对象。

对象引用主要有四种引用方式，强引用，弱引用，软引用，虚引用。

GC主要有两种方法来进行判断对象的存活：引用计数算法， 可达性分析算法（GC root）。

引用计数算法：当每个对象被创建的时候，然后有对象引用他的时候，计数器就会+1，任何引用计数为0的对象，都可以被回收。

可达性分析算法：通过一个名为GC Root的对象作为起点，寻找对应的引用节点，找到这些引用节点后，继续向下寻找引用节点。当一个对象不存在与这个寻找的引用链中时，就说明这个对象可以被回收。

垃圾收集算法分为：标记清除，标记整理，分代收集（***Copying算法***）。

标记清除：先标记出可以被回收的对象，在标记完成后统一回收这些对象，标记是可达性分析算法标记的。

优点：不需要移动对象，仅对标记的对象进行处理，在标记对象较少的情况下极为高效。

缺点：效率低，同时会产生大量的内存碎片

标记整理：先标记出可以被回收的对象，然后让所有的对象向一端移动，然后直接清理边界处的内存。

优点：经过整理后，对象的分配只需要指针碰撞移动就可以完成，内存不再存在碎片化的问题。

缺点：需要移动大量的元素，消耗时间长，效率低下。

分代收集（copying算法）：

优点：标记和复制阶段可以同时的进行，并且每一次只对一块内存进行回收，只需要移动栈顶指针就可以实现内存的分配。也不会出现内存碎片的问题。

缺点：需要一块能够容纳所有对象的内存空间，可以一次性分配的空间变少了。

1. 说下final，finally，finalize有什么区别

Final主要是用来修饰类，方法，属性，。描述的属性是常量不可以改变，描述的方法不可以被重写，Final描述的类不可以被继承

Finally 主要是用来在try catch里面使用。Finally里面放置的是一定会被执行的代码。、

Finalize，object的方法，是用来进行垃圾回收的。

1. Error和Exception的区别，try{}里有一个return语句，那么紧跟在这个try后的finally{}里的代码会不会被执行，什么时候被执行，在return前还是后?

Error是Throwable异常抛出的子类，在程序编译的时候出现的错误，所包括的是严重的系统错误，不应该被捕获处理，如内存溢出，栈溢出等。

Exception是Throwable异常抛出的子类，所包括的是

编写检查时异常 会在编写时直接报错，

运行时异常 RuntimeException 。

这些异常是可以被捕获的，并进行相应的处理，

捕获的方式分为两种：try-catch-finally 和 直接抛出异常

同时异常是可以直接自己创建的。

在return执行完成后，会在return后执行finally代码

1. **说说同步和异步的区别，多线程的几种实现方式，和run方法可以启动线程吗**

**同步：多个线程访问同一块资源时，只有当前的线程访问结束后，其他线程才可以开始访问，同时会在同步锁的地方等待运行。**

**异步：同时运行多个线程解析用户的任务，同时运行多个不同的用户任务。同时和其余的线程，一起运行。**

**Thread**

**Runnable**

**Callable**

**ThreadPoolExecutor**

**New Thread(new 任务).Start()**

**ThreadPoolExecutor.Submit(new 任务)，提交需要返回值的任务**

**ThreadPoolExecutor.execute(new 任务)，提交不需要返回值的任务，返回一个future的对象**

**Executors 返回线程池对象的弊端如下：**

**FixedThreadPool 和 SingleThreadExecutor ： 允许请求的队列长度为 Integer.MAX\_VALUE,可能堆积大量的请求，从而导致OOM。**

**CachedThreadPool 和 ScheduledThreadPool ： 允许创建的线程数量为 Integer.MAX\_VALUE ，可能会创建大量线程，从而导致OOM**

**通过Executor 框架的工具类Executors来实现 我们可以创建三种类型的ThreadPoolExecutor：FixedThreadPool ： 该方法返回一个固定线程数量的线程池。该线程池中的线程数量始终不变。当有一个新的任务提交时，线程池中若有空闲线程，则立即执行。若没有，则新的任务会被暂存在一个任务队列中，待有线程空闲时，便处理在任务队列中的任务。**

**SingleThreadExecutor： 方法返回一个只有一个线程的线程池。若多余一个任务被提交到该线程池，任务会被保存在一个任务队列中，待线程空闲，按先入先出的顺序执行队列中的任务。**

**CachedThreadPool： 该方法返回一个可根据实际情况调整线程数量的线程池。线程池的线程数量不确定，但若有空闲线程可以复用，则会优先使用可复用的线程。若所有线程均在工作，又有新的任务提交，则会创建新的线程处理任务。所有线程在当前任务执行完毕后，将返回线程池进行复用。**

1. **说说你对线程的几种状态的了解**

**新建，就绪，执行，死亡，阻塞**

**创建：线程创建后，处于新建状态**

**就绪：调用run方法，并且已经准备好，随时进入CPU运行**

**执行：得到CPU的执行时间片，开始运行**

**死亡：任务执行完成，线程停止**

**阻塞： 阻塞相对于执行状态而言，CPU执行过程中执行时间片结束，会让线程离开CPU，此时线程就处于阻塞状态。然后在线程执行的时候，sleep和wait方法均会使**

**线程直接进入阻塞状态，然后进入就绪状态。Wait（）需要notify方法才可以回到就绪状态。sleep方法，在时间结束后，直接回到执行状态。**

1. **说说你对多线程锁机制的理解**

**Synchronized 解决的是多个线程之间访问资源的同步问题，Synchronized保证被其修饰的代码块或者是方法在任意时刻只能有一个线程执行。**

**锁机制出现的原因主要是为了解决多线程并发带来的安全问题。保证多个线程执行同一块的代码的时候，这个代码只执行一次。银行取钱问题，上厕所问题**

**同步锁的钥匙在静态方法中是 当前类的.class，在普通方法中是this，分别类锁和对象锁**

1. **反射获取class对象的方式有哪些，反射创建对象的方式有哪些。**

**Person.class**

**Person.getClass()**

**Class.forName(com.bianyiit.beans.Person)**

1. Java常用的设计模式有哪些（**工厂模式，单例模式，装饰者模式**等）

工厂设计模式：当有多个类继承一个接口的时候，新建一个工厂类用来创建对象。接收的是args[]数组的第一位元素，该元素就是需要调用的类，然后就会调用工厂类生成对象

单例设计模式：

饿汉式：首先将构造方法私有化，然后建立一个私有的静态的常量用来创建对象，然后写一个公开的静态的方法用来向外输出这个对象。

懒汉式：首先将构造方法私有化，然后建立一个私有的静态的常量，然后写一个公开的静态的方法用来创建对象，并且输出对象。但是要对这个创建对象的方法进行上锁等限制。在调用这个方法的时候，判断是否创建对象，

代理模式：

1. **UDP和TCP协议的区别**

**UDP是无连接的网络发送，直接发送到端口。TCP是面向连接的，必须经历三次握手，才可以连接成功。并且还有四次回收才可以断开连接。**

**UDP是不安全的，不可靠的链接方式，所以UDP发送数据不保证可靠的交付。TCP是安全的，可靠的链接方式，必须建立链接才可以发送文件。**

**UDP的效率比TCP更高，到是所需要消耗的资源比TCP少，因为需要处理的东西更少。TCP需要处理握手和挥手。**

**TCP的链接是一对一的，UDP的链接是一对多的，多对一，多对多的**

**16、说说你对面向对象的理解**

**面向对象：object –orient-programming，面向对象的程序设计，他在程序设计中可能包含属性，方法，等。对象是类的实例，他讲对象作为基本单元，将数据和程序封装在里面，以提高程序的重用性，灵活性和扩展性。**

**面向对象一般有以下特点，**

**1，类与对象：类抽象了一件事物的，类中包含了数据的形式和数据的操作。然后对象就是类的实例、**

**2，封装：隐藏实现的细节，明确标识出允许被外部使用的所有的成员函数和数据项，**

**3，继承：子类继承父类，拥有父类的所有功能，并且可以在子类中进行扩展，实现代码的重复使用。子类和父类是兼容的，外部调用无需注实现的细节。**

**4，多态：一个接口有多个子类或者实现类，在运行编译的时候，才决定所引用对象的实际类型，再根据其实际的类型调用其对应的方法。重载也是多态的一种实现。**

17、如何避免死锁

死锁是一种可能出现的情况，并不一定会出现，当死锁出现时往往是高负载的情况下出现的。设置优先级，必须执行一个，然后再执行一个，

**18、说说冒泡排序的原理，选择排序的原理，二分查找的原理**

**冒泡排序，每一次运行都将一个最大的放到最后**

**先拿出第一个和每一个进行比较，**

**选择排序，每一次运行都将一个最小的放到最前面。**

19、wait()和sleep()的区别

Wait方法，在调用的时候，必须是在同步代码块中，在执行的时候，释放监视锁对象的器，会释放当前线程的锁对象，下一次进行的时候只能够再次重新启动线程。

Sleep方法，再任何位置都能够运行，他是让当前线程处于休眠状态，休眠时间到了，线程就会进入就绪状态，重新申请时间片。Sleep方法执行后不会释放监视锁对象的监视器，可以继续运行，

20、为什么要使用泛型以及好处

泛型：就是一种程序设计语言的特性，主要是参数化泛型

主要使用的时候：在单列和多列的集合中，提高类型的安全。在编译的时候就检查类型安全，保证类型转换的时候不会出错。减少程序运行的时候类型的转换，同时能够提升程序运行效率。

引入泛型的好处是：安全可靠，提高类型的安全，同时消除源代码中的许多强制类型的转换，使得代码更加可读，减少出错的机会。

21、什么是java的序列化，如何实现java的序列化

序列化是将对象转换成字节流，以便于长久的存储的过程。他可以保存对象的状态和数据，方便在特定的情况下，重新构建该对象。使用Serializable

需要在类中手动创建一个属性，serialVersionUId

22、hashSet和TreeSet的区别

Hashset和Treeset都是set接口的实现。

HashSet是无序的set表，基于hash表实现，

TreeSet是基于二叉树的实现。需要实现comparable接口

23、java中io流有哪些，各有什么特点和功能

字节流：字节输入流，字节输出流。

字符流：字符输入流，字符输出流。

缓冲流：缓冲字节流，缓冲字符流：先在内存中提供一个缓冲区，先从磁盘从读取一部分文件至缓冲区，然后cpu直接操作内存，极大提高输入和输出的效率，因为减少了很多与硬盘的操作。

字符与字节互转流：

InputStreamReader是从字节流流向字符流的桥梁：它读取字节，并使用指定的charset将其解码为[字符](mk:@MSITStore:D:\\QMDownload\\jdk%20api%201.8_google\\jdk%20api%201.8_google.CHM::/java/nio/charset/Charset.html" \o "java.nio.charset中的类) 。

OutputStreamWriter是字符流流向字节流的桥梁，向其写入的字符编码成使用指定的字节[charset](mk:@MSITStore:D:\QMDownload\jdk%20api%201.8_google\jdk%20api%201.8_google.CHM::/java/nio/charset/Charset.html) 。：向其写入的字符编码成使用指定的字节[charset](mk:@MSITStore:D:\\QMDownload\\jdk%20api%201.8_google\\jdk%20api%201.8_google.CHM::/java/nio/charset/Charset.html" \o "java.nio.charset中的类) 。

标准输入输出流：PrintWrite，PrintRead

对象操作流：对象字节输入流，对象字节输出流，需要在类中手动指定一个属性---serialVersionUID---版本号。

**24、hashmap底层实现原理**

**25、ArrayList底层实现原理**

**1，底层实现原理是数组，只是可以动态的自动扩容。**

26、说说你对java虚拟机的了解

1，JVM的内存结构：方法区，堆内存，栈内存，本地方法栈，程序计数器。

JVM是java一次编译多处运行的核心。Java编译器只需要面向JVm生成JVM可以理解的代码或者解释文件，在特定的平台上，只要安装了对应的JVM，就可以在上面执行JAVA的字节码程序。

Java的GC垃圾回收器：

27、&和&&的区别

&在执行的时候，必须判断完所有的条件才会结束。

&&在执行的时候，判断出现一次false就会结束。

28、==和equals的区别

==和equals 在object的类里面，都是比较内存地址。先判断HashCode，然后判断equals

==和equals 在string的类里面，==判断的是内存地址，equals是把String转换成char[]，然后进行对比每一位的内容是否相等。

然后在其他的比较的时候，

29、String s = new String("xyz");创建了几个String Object? 二者之间有什么区别？

会创建两个object的对象，会产生多余的对象，也就是内存垃圾。

第一个对象是“xyz”产生的，第二个对象是new String产生的。

**30、线程池的核心类是哪一个？有几种线程池？分别有什么区别？**

**31、说一说IO和NIO的区别以及应用场景？**

**32、1.8的新特性你知道哪些？**

**1，**对接口里面的方法，可以进行具体的实现，但是这个方法必须是default，子类可以直接使用这个方法，同时子类中可以对这个方法进行重写。**当同时接收继承父类函数，同时继承接口的时候，会优先实现父类中的函数。**

**2，重复的注解**

**3，**Clock类 4，对于Base64的支持，新的Base64API也支持URL和MINE的编码解码。

5，HashMap当总容量大于64，碰撞的元素个数大于8时，会从链表转变成红黑树

**32、1.7的新特性你知道哪些？**

**1，数值中可以使用下划线链接，编译的时候会直接舍弃，不会进入程序。并且位置可以随意指定**

**2，swtich可以支持字符串类型**

**3，AotoCloseable接口，可以自动的停止相关的东西，可以不写Finally**

**4，泛型的自动搜索，List<String> list = new ArraryList<>();**

33、java中会存在内存泄漏吗，请简单描述

内存泄露：程序在申请内存后，无法释放已申请的内存空间，

原因：长生命周期的对象 含有 短生命周期的对象的引用。尽管短生命周期的对象已经不再需要，但是因为被长生命周期的对象持有，所以也无法释放。

1. 长生命周期的静态集合类 所 含有的一些短生命周期的对象，如循环插入匿名函数对象。GC无法回收这些匿名函数对象。
2. 代码在死循环中一直创建对象

内存溢出：程序在申请内存时，没有足够的内存空间。

JVM虚拟机的内存是有限的，在死循环的时候，一直申请创建对象会造成JVM虚拟机的内存的溢出。

34、说一说常用的协议有哪些?

35、说一说你了解的数据结构。

ArraryList，LinkList，AVL树较深了解，Hashmap简单了解。

36，堆内存的分配区域

1. 年轻代，与新生代GC
2. 年老代，与老年代GC
3. 持久代