首先JVM面试一般逃不过：GC算法，类加载机制，调优（堆内存设置，算法GC算法配置）

我们同时站在面试者与被面试者去阅读深入java虚拟机，去思考自己会去考察什么，通过搜罗面试题以及有经验的大神们比较关注什么

第一章：

相当于引言，如果是我我是不会浪费时间在这个上面

第二章：Java内存区域与内存溢出异常

1作者以运行时数据区域来对标题呼应，java内存区域和运行时数据区域是一个意思么？（方法区、本地方法栈，程序计数器，堆，栈+直接内存）

2.程序计数器为什么没有规范OOM？？OOM是在什么情况下发生的？或者问程序计数器在什么情况下会抛出OOM异常？（好像有点误导的感觉）

在程序运行时内存空间达不到需求时就会抛出OOM，程序方法所需要的栈深度超过虚拟机分配的深度后会抛出Stack Overflow，一般虚拟机都会支持栈的动态扩展，一旦扩展不了的时候就会抛出OOM

3.java规范对虚拟机栈规定了两种内存异常？他们在什么情况下发生?

两种基本在第2问讲过

4.堆主要存放的是什么数据？（很多人可以答出所有的对象的实例，但是会漏掉数组不是原始类型的数组）

**5.JIT编译器有没有？逃逸分析技术呢？栈上分配，标量替换？所有的对象一定都在堆上么？**

6为什么要优化呢？在堆上不好么？堆的空间那么大，栈那么小？

堆上内存大事固然不错的，但是无法自动回收，需要依赖GC，GC在大多数时候很容易引起性能问题，但放在栈里就不用考虑垃圾回收的问题，但是您也说了，栈空间很小，在需要栈上分配的对象必须谨慎，否则也逃不过stack或者OOM

7**栈上分配技术有没有了解过**

8**标量替换有没有了解过**

9你用过哪些虚拟机（HotSpot VM、J9 VM、Zing VM，BEA Jrockit）

10方法区主要放置什么（那些不怎么变的，比如类信息，常量，jit编译的代码（马长累）），常被人成为permanent，注意姿势大家这么说，并非官方说法。因为GC有的时候还是会拜访这个区域的，只是一般都很难访问，甚至有的虚拟机都没有！

11除了刚刚说的这些内存区域，还有没有其他的？（直接内存，）

12直接内存是一个容易被忽略的boy，一般虚拟机配置内存的时候都不会将其包含进去，然而其确实又占内存，很容易出现个个内存区域总和大于物理内存限制，导致OOM

13.直接内存一般用来（NIO）

14句柄和指针（句柄的本质仍然是指针，但是指针的权限因为太大，所以就进行了封装，就像身份证证和学生证的区别吧）

15引用是用什么实现的（或者说有那两种实现方式）：使用句柄的好处在于，当对象被移动的时候，引用本身不会修改，缺点就是慢，使用指针的方式就反着看。

**16有没有遇到OOM或Stack Overflow的的情况，考察排查这两种问题的方法**

## 垃圾回收专题

**判活算法：引用计数算法和跟搜索算法**

引用计数算法是当每当对象被引用是就会将其引用计数加1，判断对象dead的方法自然也就是看引用计数是否为零，跟搜索算法可以理解为一个应用链专业一点就是图，对象到GC root总是可以找到路可走，这种判活的方法不仅可以判断某一个，还可以判断多个相互引用但无法追踪到gc root的多个对象。关于这两种算法，java选择的后者。至于为什么，其实我认为不需要过于纠结，两种算法各有利弊，而且有些编程语言也会选择引用计数法。不同取舍罢了。而且真正落实到垃圾回收层面，参考不仅仅是生死问题。

Q：GC root是什么

A:总结为4种，虚拟机栈中引用的对象（本地变量），方法区中的类静态属性引用，方法区的常量引用的对象，本地方法栈中引用的对象（为毛是这些？？？ gc root使用原则就是活跃的引用，对于java程序来说能够活跃一般都是静态变量，常量，方法使用的变量（java方法，本地方法））

Q:两次标记

A:jvm其实还是比较仁慈的，通过跟搜索算法找到的不可达对象并不是立即处以死刑，而是先标记一下，在标记的时候，会筛选一下，赛选条件就是对象没有覆盖finalize方法或者finalize方法已经被执行过，而且就算判断为有必要执行该方法，也是放到F-queue队列中，由一条低优先级的线程去执行销毁行为，也就是说这个线程也有可能不执行，这个时候如果想要重新激活对象还是有可能的，再往后就会进行第二次标记，在第二次标记前如果没有重新激活，可能就在劫难逃。

Q:四种引用类型

A:强，软，弱，虚 ==（亲情，爱情，友情，一面之缘）SSWP

只要强在，就不会回收，软引用会在内存不够用时第二次回收进行回收，弱引用和虚引用基本上只要进行gc就会被kill掉，虚引用连对象都获取不到。似乎可以通过改变引用来认为控制对象的生存能力，也算为通过编码的方式控制对象是否可以被回收提供的方法。

Q:方法区的回收

A：方法区虽说是持久代，但不代表不回收，只是说相对长寿一些。一般回收的是不用的常量和无用的类。