总复习-JVM认知

JDK,JRE

生命周期:启动，执行，结束

运行时数据区域:虚拟机栈，本地方法栈，程序计数器是线程私有的,方法区和堆是线程共有的

堆的空间是不是一定是连续的，java虚拟机官方规范中没有要求堆一定是连续的，而大多数虚拟机为了开发 维护容易写成可连续的

程序计数器:一个只读的存储器，用于记录Java虚拟机正在执行的字节码指令的地址

Java虚拟机栈：线程私有的存储器，每个⽅法在执⾏的同时都会创建⼀个栈帧（StackFrame）⽤于存储局部变量表、操作数栈、动态链接、⽅法出⼝等信息

本地方法栈：本地方法栈是一种特殊的栈，他和Java虚拟机栈有着相同的功能，但是它支持本地代码的执行，本地⽅法栈中就是C和C++的代码

堆（里面包含字符串常量池）：是存储对象实例，所有的对象实例以及数组都要在堆上分配。

方法区（永久代）：用于存储已被加载的类信息，常量，静态变量，即时编译后的代码等数据的内存区域

运行时常量池：存在于方法区中，用于存储编译生成的信息，主要有字面量和符号引用常量两种

直接内存和堆内存的区别

直接内存申请空间耗费很⾼的性能，堆内存申请空间耗费⽐较低

直接内存的IO读写的性能要优于堆内存，在多次读写操作的情况相差⾮常明显

分配对象的两种方式：指针碰撞空闲列表

垃圾回收机制（GC）

final，finally，finallize（）区别

final 修饰的成员变量，必须在声明的同时赋值，⼀旦创建不可修改（常量）；修饰的⽅法，不能被⼦类重写；修饰的类不能被继承；

finally 是异常处理语句结构 的⼀部分，不管是否有异常，他的语句总是执⾏。 ⼀般⽤来存放⼀些关闭资源的代码。

finalize 是Object类的⼀个⽅法，主要是在收集器执⾏的时候会调⽤被回收对象的此⽅法，进⾏资源回收，例如关闭⽂件等。

新生代，老年代（1：2）

新⽣代 ( Young ) 被细分为 Eden 和 两个 Survivor 区域（8：1：1）JVM 每次只会使⽤ Eden 和其中的⼀块 Survivor 区域来为对象服务，所以⽆论什么时候，总是有⼀块 Survivor 区域是空闲着的

Minor GC、Major GC、Full GC

区别

1. Minor GC是新⽣代GC，指的是发⽣在新⽣代的垃圾收集动作。当Eden没有⾜够空间的时候就会触发jvm发起⼀次Minor GC。由于java对象⼤都是朝⽣夕灭的，所以Minor GC⾮常频繁，⼀般回收速度也⽐较快

2. Major GC是⽼年代GC，指的是发⽣在⽼年代的GC，通常执⾏Major GC会连着Minor GC⼀起执⾏。Major GC的速度要⽐Minor GC慢的多。

3. Full GC是清理整个堆空间，包括年轻代和⽼年代

Minor GC触发条件

当申请⼀个对象时，发现eden区不够⽤，则触发⼀次MinorGC。

Major GC和Full GC 触发条件

1. 每次晋升到⽼年代的对象平均⼤⼩>⽼年代剩余空间

2. MinorGC后存活的对象超过了⽼年代剩余空间

3.老年代空间不⾜

4. 执⾏System.gc()（有可能会触发，也有可能是触发Minor GC）

5. 堆内存分配很⼤的对象

JVM如何判断对象是否存活

•引用计数法：给对象中添加一个引用计数器，每当有一个地方引用它，计数器就加1；当引用失效，计数器就减1；任何时候计数器为0的对象就是不可能再被使用的。这个方法实现简单，效率高，但是目前主流的虚拟机中并没有选择这个算法来管理内存，其最主要的原因是它很难解决对象之间相互循环引用的问题。

可达性分析算法：这个算法的基本思想就是通过一系列的称为，"GC Roots”的对象作为起点，从这些节点开始向下搜索，节点所走过的路径称为引用链，当一个对象到 GC Roots 没有任何引用链相连的话，则证明此对象是不可用的。