题目 01- 请你用自己的语言向我介绍 Java 运行时数据区（内存区域）

笔记本:	My Notebook		
创建时间:	2023/4/24 9:48	**更新时间:**	2023/4/24 12:19
作者:	q05xt3eo		
URL:	https://u.geekbang.org/lesson/498?article=634043&utm_source=time_web&ut...		

题目 01- 请你用自己的语言向我介绍 Java 运行时数据区（内存区域）

1.堆： 主要负责引用对象的存储

2.虚拟机栈： 包含动态链接 ，方法返回地址，局部变量表，操作数栈。

3.本地方法栈：负责本地方法的调用

4.元空间(方法区)： 主要存储类信息

5. 运行时常量池： 保存自变量和符号引用

6. 直接内存：直接申请的内存，适合读写频繁场景

题目 02- 描述一个 Java 对象的生命周期

1. 一个对象的创建过程
   01.调用new指令
   02.检查常量池中是否存在类的符号引用，不存在则先进行类加载
   03.检查该类之前是否被加载过
   04.分配内存空间 两种分配方式 （1.指针碰撞 2.空闲列表）保证线程安全方式： （1.TLAB:为线程预分配私有空间 2.CAS）
   05.内存空间初始化零值
   06.对象头等必要参数设置
   07.执行init初始化
2.一个对象的内存分配
   1. 新对象大多数都默认进入新生代的Eden区
   2.对象进入老年代的四种情况
      01.大对象直接进入老年代，防止在新生代频繁复制效率不高(前提：前提是Serial和ParNew收集器)
      02.存活年龄过大，默认15【 -XX:MaxTenuringThreshold 】
      03.动态年龄判断，当minorGc后，发现survivo区有一批对象所占空间大于总空间的50%时，会将年龄大于年等于这批对象最大龄的对象放入老年代。
      04,.minorGc后survivo区容纳不下
3.一个对象的销毁过程
      1.通过分析对象引用链判断是否为垃圾对象，
         分析方式： 引用计数法（Reference Counting） 当这个对象引用都消失了，消失一个计数减一，当引用都消失了，计数就会变为0。此时这个对象就

会变 成垃圾。（有循环引用问题）

2.根可达算法： 从Gcroots节点开始向下搜索 ，当一个对象到GCRoot没有任何引用链相连时，则证明此对象 是不可用的，也就是不可达的。

3.回收过程分为两次标记，第二次成功的对象会被回收

第一次标记：如果对象可达性分析后，发现没有与GC Roots相连接的引用链，那它将会被第一次标 记；

第二次标记：第一次标记后，接着会进行一次筛选。筛选条件：此对象是否有必要执行 finalize() 方法。在 finalize() 方法中没有重新与引用链建立关联关系的，将被进行第二次标 记。

4.对象的 2 种访问方式是什么?

1.句柄访问

2。直接引用访问

5.为什么需要内存担保?

因为新生代采用的是复制算法，无法保证minorGc后存货的对象一定小于survivo的容量，当大于survivo容量时此时需要进入老年代担保，保证对象能顺利进入老年代，当老年代空间不足时会触发FullGc回收老年代

题目 03- 垃圾收集算法有哪些？垃圾收集器有哪些？他们的特点是什么?

1. 常见垃圾收集算法有

01。标记清除

效率不高，标记和清除过程的效率都不高 空间碎片，会产生大量不连续的内存碎片，会导致大对象可能无法分配，提前触发GC 。

02. 复制算法

优点：没有碎片化，所有的有用的空间都连接在一起，所有的空闲空间都连接在一起

缺点：存在空间浪费

03. 标记整理

优点：没有碎片化，所有的有用的空间都连接在一起，所有的空闲空间都连接在一起

缺点：性能较低，因为除了拷贝对象以外，还需要对象内存空间进行压缩，所以性能较低

2.垃圾收集器

01. **Serial收集器：年轻串行**

特点：单线程收集，标记复制算法.适合单核cpu，没有线程切换消耗

02。 Serial Old收集器：老年串行

特点：搭配serial New使用 ， 采用标记整理算法

03. Parallel Scavenge

特点： 吞吐量优先收集器 新生代使用并行回收收集器，采用复制算法 老年代使用串行收集器

04. Parallel Old

特点： Parallel Scavenge收集器的老年代版本，使用多线程和“标记-整理”算法。 在注重吞吐量，CPU资源敏感的场合，都可以优先考虑Parallel Scavenge加Parallel Old收集器。

05。 ParNew

特点 ： 新生代并行（ParNew），老年代串行（Serial Old）Serial收集器的多线程版本

06. CMS收集器

特点： 低延迟：减少STW对用户体验的影响【低延迟要求高】 并发收集：可以同时执行用户线程 CMS收集器不能像其他收集器那样等到老年代

几乎完全被填满了再进行收集，而是当堆内存使用率 达到某一阈值时，便开始进行回收。 CMS收集器的垃圾收集算法采用的是标记-清除算法。 会产生内存碎片，导致并发清除后，用户线程可用的空间不足。 CMS收集器对CPU资源非常敏感。

07. G1（Garbage-First）收集

特点： 1. 并行与并发：充分利用多核环境下的硬件优势 2. 多代收集：不需要其他收集器配合就能独立管理整个GC堆 3. 空间整合： “标记-整理”算法实现的收集器，局部上基于“复制”算法不会产生内存空间碎片 4. 可预测的停顿：能让使用者明确指定消耗在垃圾收集上的时间。代价是回 收空间的效率降低。