题目 01

请你用自己的语言介绍 Java 运行时数据区（内存区域）

1.堆、虚拟机栈、本地方法栈、方法区（永久代、元空间）、运行时常量池（字符串常量池）、直接内存

堆：存放java对象的地方，占据jvm最多的内存，是垃圾回收主要清理的空间，当堆空间装不下会oom，如果创建的对象无法被清理会造成内存泄漏，多次内存泄漏最终会导致oom。

虚拟机栈：控制程序方法调用的顺序，每个线程都有自己的栈空间，所以栈是线程安全的。

本地方法栈：与虚拟机栈类似，虚拟机栈调用的是java服务，本地方法栈调用的是native方法，因为java无法直接进行操作系统系统调用，需要借助c++方法去实现系统功能的调用

方法区：两种实现永久代和元空间，永久代存放的是运行时常量池，类型信息，类变量，class实例的引用，方法表等，元空间只存类的元信息，元空间把静态变量和运行时常量放到了堆中，降低了内存溢出和性能问题，同时永久代使gc变得复杂且回收效率低，永久代用的是jvm的内存空间，元空间用的是物理内存空间

运行时常量池：常量池分为静态常量池和运行时常量池，字符串常量池从运行时常量池里面单独分出来是因为字符串是不可变量，一旦创建就无法更改，字符串占用内存较大需要单独分配空间提高查找效率，运行时常量池分为字面量和符号引用，字面量存放的是基本数据类型，符号引用存放的是class对象，方法和属性。

直接内存：jvm以外的物理内存，为native方法提供内存空间，直接内存分配空间更消耗性能，读写要优于堆内存

2.为什么堆内存要分年轻代和老年代?

堆内存空间有限，会产生很多垃圾需要被清理，但是不同对象的生命周期不一样，需要分批清理，但是垃圾过多会导致无法创建新对象，因此需要把未被清理的老对象整理好，方便空出空间来存放新对象。

题目 02

描述一个 Java 对象的生命周期

1.解释一个对象的创建过程

当一个对象被new出来时，会首先去常量池中查找类的符号引用，如果不存在会重新加载类，如果已加载会为对象分配内存空间，然后初始化空间，在对象头加入必要信息（属于哪个类，类的位置，对象的哈希码等），最后会数据初始化

2.解释一个对象的内存分配

对象的分配分为指针碰撞和空闲列表，指针碰撞是分配连续的内存地址，空闲列表分配不连续的内存地址

3.解释一个对象的销毁过程

一个对象在新生代被创建，当新生代满的时候，垃圾回收线程会扫描该对象是否被引用，有引用的对象会被放入幸存区，超过15次未被清理的对象会放入老年代，未被引用的对象会被垃圾回收

4.对象的 2 种访问方式是什么?

句柄和直接指针

5.为什么需要内存担保?

幸存者区比较小，如果新生代存活的对象较多，幸存者区装不下，就需要老年代进行内存担保来把无法容纳的对象提前加入老年代，每次内存回收存活下来的对象是未知的，只好取每次垃圾回收进入老年代的对象大小平均值和老年代剩余空间比较，决定是否full gc来让老年代空出更多空间

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.ParNew 收集器

Serial收集器的多线程版本，新生代是并行收集，老年代是串行收集，单cpu不如Serial，因为多线程需要上下文切换，会增加系统开销

2.ParallelScavenge 收集器

目标是达到一个可控制的吞吐量，新生代使用并行收集，老年代使用串行收集，

3.ParallelOld 收集器

Parallel Scavenge收集器的老年代版本，新生代和老年代都是并行收集，在注重吞吐量，cpu资源敏感的场景，都可以考虑Paraller Scavenge加Parallel Old

4.CMS 收集器

低延迟，减少stw对用户体验的影响，可以同时执行用户线程，cms不像其他收集器要等到老年代几乎填满再收集，而是当堆内存使用到达阈值就开始回收，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，对cpu资源非常敏感

5.G1 收集器

面向局部收集，充分利用多核环境的硬件优势，不需要其他收集器配合就能独立管理整个gc堆，不会产生内存碎片，能让使用者明确指定消耗在垃圾收集上的时间，更短的gc回收和回收空间效率低