自我介绍：

名字，在\*\*\*\*组，从事RMQ相关工作。

16年本科毕业，\*\*\*\*大学，壹钱包，大四实习1年，工作一年。

中间件相关工作

问题1：创建线程和启动线程的方法？Start和 run区别？

回答：New thread，thread（runnable），线程工程，线程池，没有显示创建线程。

【答案】：多线程实现方式主要有四种：继承Thread类、实现Runnable接口、实现Callable接口通过FutureTask包装器来创建Thread线程、使用ExecutorService、Callable、Future实现有返回结果的多线程。

其中前两种方式线程执行完后都没有返回值，后两种是带返回值的。

调用start方法方可启动线程，而run方法只是thread的一个普通方法调用，还是在主线程里执行。用start方法来启动线程，真正实现了多线程运行，这时无需等待run方法体代码执行完毕而直接继续执行下面的代码。通过调用Thread类的 start()方法来启动一个线程，这时此线程处于就绪（可运行）状态，并没有运行，一旦得到cpu时间片，就开始执行run()方法，这里方法 run()称为线程体，它包含了要执行的这个线程的内容，Run方法运行结束，此线程随即终止。

问题2：若某二叉树的前序遍历访问顺序是abdgcefh，中序遍历访问顺序是dgbaechf，则其后序遍历的结点访问顺序是？

回答：正确

先找到头结点，然后分别识别出左子树和右子树，

程序，先创建二叉树，两个数组，start1,2和end1,2

【答案】：

A

/ \

B c

/ / \

D e f

\ /

G h

（gdbehfca）

问题3：java的内存模型了解吗？

回答：程序计数区（下一步执行的字节码），虚拟机栈（线程私有），本地方法栈（局部变量，方法返回），方法区（和厂商相关，常量池，类编译字节码），metaspace，堆（创建对象，有垃圾回收），堆外（需要主动回收垃圾）

Volatile和synchronize

【答案】：

主内存和工作内存，实例字段，静态字段，构成数组对象的元素，不包括局部变量和方法参数。

内存间交互操作：lock,unlock,read,load,use,assign,store,write

另一个角度：java虚拟机运行时数据区

程序计数器：小内存空间，当前线程所执行的字节码的行号指示器，线程私有，没有OOM

java虚拟机栈： 线程私有，java方法执行的内存模型，栈帧，存储局部变量表、操作数栈、动态链接、方法出口等。局部变量表存放编译期可知的基本数据类型、对象引用、和returnAddre类型。线程请求的栈深度过大，StackOverflowError；动态扩展栈无法申请到足够的内存，OOMError。

本地方法栈：虚拟机使用到的native方法

java堆： 线程共享，存放对象实例，数组。GC堆，细分为新生代和老年代，Eden空间、from Survivor空间、to Survivor空间等。物理内存可以不连续。无内存完成实例分配，并且堆也无法再扩展时，抛OOMError。

方法区：线程共享，存储已被虚拟机加载的类信息、常量、静态变量、即时编译器编译后的代码等。运行时常量池，存放编译期生成的各种字面量和符号引用。

直接内存：并不是JVM运行时数据区的一部分， 可直接访问的内存， 比如NIO会用到这部分，文件映射也会用到直接内存。