问题：

1. FairScheduler调度是否会考虑作业优先级？MR的优先级与Yarn的优先级如何对应？

2. 占用其他队列资源是否有规则？可否限制队列不能被其他队列抢占？

3. 发生抢占时，AM杀死container的策略是怎么样的？以MR为例说明

4. 举例说明FairScheduler调度过程

5. FairScheduler的计算资源权重比R的解法？

6. RM如何选择NodeManager进行调度container？

7. 节点有空闲资源时，Container是否可以超过内存限制阈值？

8. 资源抢占，选择container进行kill时，是否考虑container启动时间？什么策略？

9. 结合应用场景，详述FairScheduler与CapacityScheduler各自的优缺点？

1. FairScheduler调度是否会考虑作业优先级？MR的优先级与Yarn的优先级如何对应？
   1. 调度首先选择 队列Queue、然后选择这个队列的app，然后选择app的container请求。Container请求的优先级会有不同，客户端提交app时填的优先级是 申请运行appMaster的container请求的优先级。然后appMaster申请container时候的请求的优先级由AM自己决定。
   2. 一句话：选择队列和app没有考虑优先级，选择app的container请求考虑优先级。这样做有个很明显的好处，如果某个捣乱的AM把自己的资源请求priority都是设置地很高，也不会影响其它app的资源调度。
   3. 默认AM\_CONTAINER\_PRIOIRTY = 0, PRIORITY\_MAP = 20, PRIORITY\_REDUCE = 10。数值小的优先级高。
   4. 在调度器里面，会为每个app用一个AppSchedulingInfo的对象表示，这个对象保存了app的所有申请资源的请求和每个请求的优先级。选择了某个app后，根据资源请求的优先级来进行调度。
2. 占用其他队列资源是否有规则？可否限制队列不能被其他队列抢占？
   1. 分两部分讲。
   2. (1)占用其他队列的资源，这里没有抢占。CapacityScheduler和FairScheduler都会实时计算每个队列这一刻应得的资源量。虽然可以给每个队列配置最小资源量，但这并不是硬性保证，只要它暂时不用这些资源，就可能被其他队列占用。就是说，最小资源量只用用来计算队列应得资源量的一个变量。
   3. (2)默认都不开启抢占，只有配置文件设置开启时才会抢占。而且别抢占的队列被抢占资源后，剩余的资源量不会少于它的应得资源量。
   4. 重点就是 每个队列实时的 应得资源量。配置文件的参数可以影响应得资源量。
3. 发生抢占时，AM杀死container的策略是怎么样的？以MR为例说明
   1. 通知AM去处理，可以是AM自己去杀死container，或不作为。MRAppMaster是什么都不做。可以从代码里看出：org.apache.hadoop.mapreduce.v2.app.rm.RMContainerAllocator# getResources()方法可以看出。
4. 举例说明FairScheduler调度过程
   1. 另外说
5. FairScheduler的计算资源权重比R的解法？
   1. TotalResource = Σ Min{ max(weight[k] \* R, minShare[k]), demand[k] } k=1..n

也就是解这个方程。

1）初始化R=1.0，计算方程右边的表达式，如果小于TotalResource，R翻倍，直到大于TotalResource。

2）在R/2跟R之间进行二分查找，迭代25次，就认为找到了R。

参见：org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.policies.ComputeFairShares类

1. RM如何选择NodeManager进行调度container？
   1. 收到哪个NM节点的心跳，就在这个NM节点上分配资源。
2. 节点有空闲资源时，Container是否可以超过内存限制阈值？
   1. 不可以，不然就不叫内存资源隔离了。
3. 资源抢占，选择container进行kill时，是否考虑container启动时间？什么策略？、
   1. 考虑了container启动时间。Kill container的时候，首先选择优先级低的，如果优先级相同，选中启动较晚的作业，这样浪费的计算资源更少。
4. 结合应用场景，详述FairScheduler与CapacityScheduler各自的优缺点？
   1. 两个队列功能和调度上都差不多，还是有些差别。如果管理员想要控制某些队列能获得更多资源，而且保证集群的利用率，适合使用CapacityScheduler，它更能保证某些队列的资源量。
   2. FairScheduler基于最大最小公平算法分配资源，更多地表现为公平。队列的最小共享量越大，在集群繁忙的时候分配到的资源就越多。但是，如果每个用户都将最小共享量设置到最大，不利于集群间的资源共享。建议，将队列愿意共享出来的内存大小和队列的权值挂钩。含义就是，你愿意共享得越多，在整体资源分配中就越能分得更多的资源。
   3. 从配置文件上看，各有好处。比如想增加某个队列的资源量。使用FairScheduler管理员只需要修改这个队列的权值和最小资源量。而CapacityScheduler需要同时修改兄弟队列的资源量。另一方面，CapacityScheduler可以精确地配置某个队列使用集群多少量的资源，而FairScheduler会比较困难。所以，使用CapacityScheduler需要了解更多的集群运行信息，需要知道整个集群跑的业务和重要程度。
   4. 总的来说，个人觉得两个队列共性多于差异。CapacityScheduler偏向保证某些队列的资源量。FairScheduler保证公平性。所以跟集群的使用有很大关系。如果是多用户使用，而且不需要明显的保证某个队列资源，建议使用FairScheduler。比如，如果集群要保证60%的资源给商业用途，建议使用CapacityScheduler。