1.

1. netty怎么处理客户端或服务端异常关闭后，缓冲区的数据未及时处理造成丢失的问题，BlockingQueue是否可以运用？

连接异常关闭后，缓冲区的数据未能及时处理就只能丢弃，或者存储起来，等目标用户重连后再重新发送（像微信一样，某个用户的连接断了，等他重新连过来时重发），但是这时知道目标用户（因为连接重建后是一个新的，不能通过连接来判断要重发的目标）

2、netty怎么做到同步调用，对于客户端可以在同一channel中实现同步返回，还是需要单独建立一个用于接收返回消息的channel.

这里的同步调用是什么意思？ 因为channel本身是双向的，可读可写，服务端写数据，客户端就能够同步返回。

3、客户端消息异常重发，基于exceptionCaught异常事件处理，怎么得到异常的消息。

异常发生后，就会捕获异常信息,并调用ChannelHandler中的exceptionCaught，把异常信息传递给他。至于要实现异常重发，可以有exceptionCaught中加入重发逻辑。

K76

2.

1. selector内部是一个线程循环维护注册在其中的channel状态，还是操作系统回调selector，通知channel状态改变？

不是selector自己维护，也不是操作系统回调。Selector实际上是通过JNI调用到操作系统的select系统调用（ linux下的Epoll，或freebsd下的kqueue 等 ），由操作系统内核负责监控channel的状态改变，并返回给selector。

1. NIO任然是阻塞的，是说阻塞的从内核态到用户态吗？对应JAVA中就是从channel中读取数据到buffer吗

对，当内核态数据准备就绪，NIO select就会返回，但是数据从内核态到用户态数据拷贝仍然是阻塞的，也就是 Java中channel读取数据到buffer中的过程。

1. 为什么从内核态复制到堆内存上需要先复制到堆外用户态内存，再复制一次到JVM？

弄反了。是当要通过网络发送数据时，需要把JVM 堆内存上的数据复制到内核态，此时无法直接复制，而是需要创建一个临时的native的内存，先把数据从JVM堆上复制到临时native内存上，再从临时native内存复制到内核态进行发送。

至于为什么要有这一次中间临时复制，是因为JVM堆内存是动态的，可能会发生GC， 而GC操作会涉及到数据的移动操作，移动数据使得其地址发生变化，会导致数据复制出错。所以JVM堆内存不能直接往内核态复制。

L10