1 如果多个进程同时拥有某一个套接字，并且都对该套接字进行读写操作，那么是不是会发生紊乱？

**上机实践表明**：当TCP接收到数据时，会将数据随机递交给其中一个进程，后续接收到的数据内核都会递交给进程处理。当该进程挂掉后，TCP连接不会释放，内核会将接收到的数据再递交给剩下进程中的一个。

2 某一个套接字已经在某一个进程中进行过读写操作了，此时fork出另一个进程，并在另一个进程中也对该套接字进行读写操作，并且关闭原进程，此时套接字可以正常工作吗？

3 如果把上面的进程换成线程呢？

4 套接字应该与进程没有绑定关系，任何一个进程只要拥有套接字，那么即可以进行收发数据操作？

**是的**，因此实际应用中有下列用法：

预先创建一个进程池，客户端每建立一条连接，服务器就从该池中选出一个空闲(Idle)子进程来处理该连接。

上面的方式很高效，因为**减少了动态创建子进程的性能损耗**，反应的及时性大大增强。这里恰恰就出现了我们前面提到的问题，所有子进程都是在服务器Listen到一条连接以前就已经fork出来了，也就是说新的连接描述符子进程是不知道的，需要父进程传递给它，它接收到相应的连接描述符后，才能与相应的客户端进行通信处理。这里我们就可以使用'传递文件描述符'的方式来实现。

在'UNIX网络编程第1卷'的14.7小节中对这种技术有详细的阐述，实际上这种技术就是利用sendmsg和recvmsg在一定的UNIX域套接口(或者是某种管道)上发送和接收一种特殊的消息，这种消息可以承载'文件描述符'罢了，当然操作系统内核对这种消息作了特殊的处理。在具体一点儿'文件描述符'是作为辅助数据(Ancillary Data)通过msghdr结构中的成员msg\_control(老版本中称为msg\_accrights)发送和接收的。值得一提的是发送进程在将'文件描述符'发送出去后，即使立即关闭该文件描述符，该文件描述符对应的文件设备也没有被真正的关闭，其引用计数仍然大于一，直到接收进程成功接收后，再关闭该文件描述符，如果这时文件设备的引用计数为0，那么才真正关闭该文件设备。