计算机网络协议开发实验报告

201220064　吴浩然

一、实验目的

熟悉网络应用层协议的设计、实现，掌握服务器套接字编程。

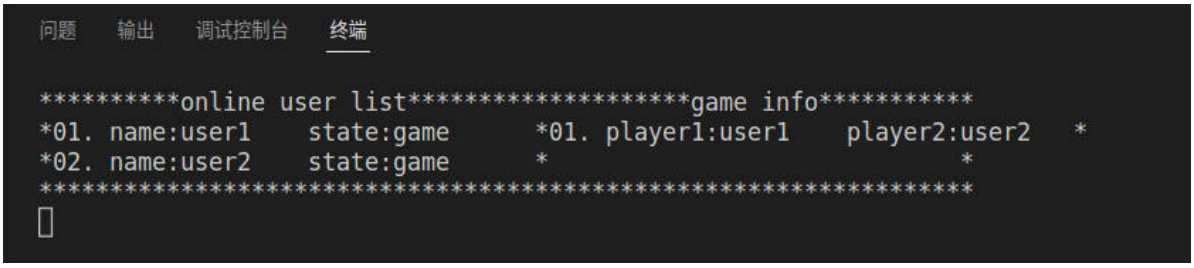
二、实验内容

功能介绍

服务器

用户状态监视功能

服务器启动后，能够从界面中实时监视用户状态及游戏对局等信息，如下图所示：



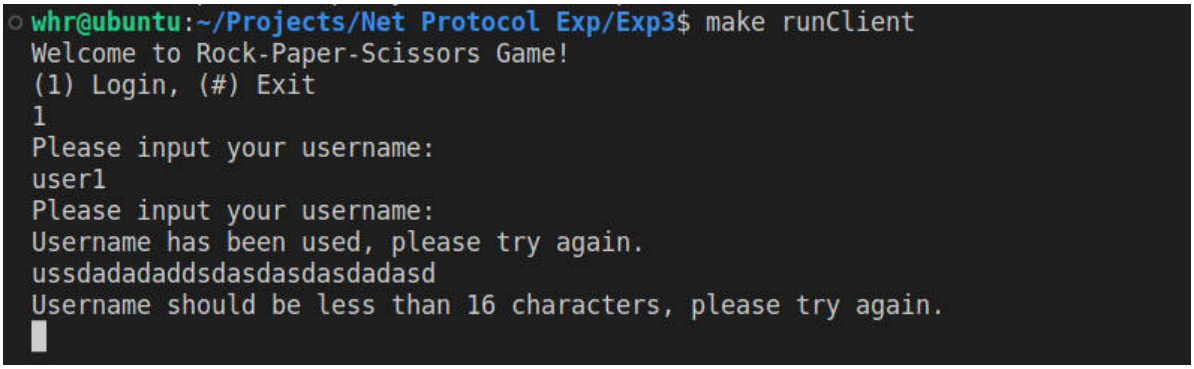
退出服务器

按“#”键退出服务器。

客户端

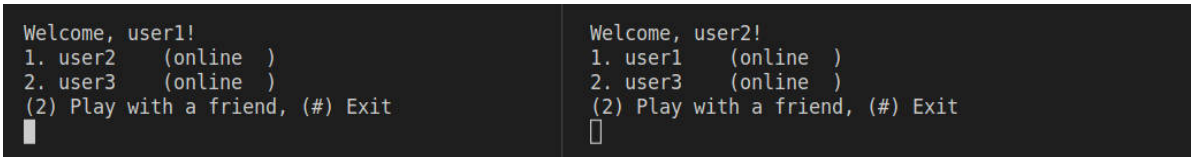
登录

客户端启动后，按“1”选择登录功能，输入用户名登录，实现了用户名长度（不允许超过15字）和重复检查，如下图所示：



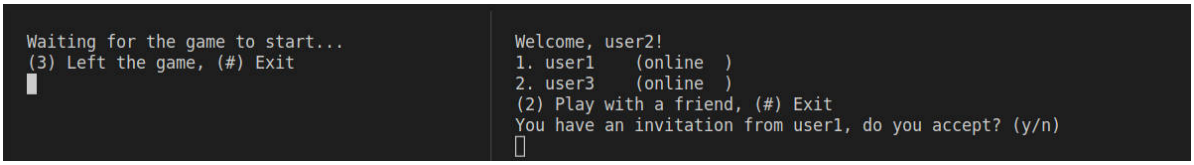
显示好友列表

登录后显示好友列表，实时更改：



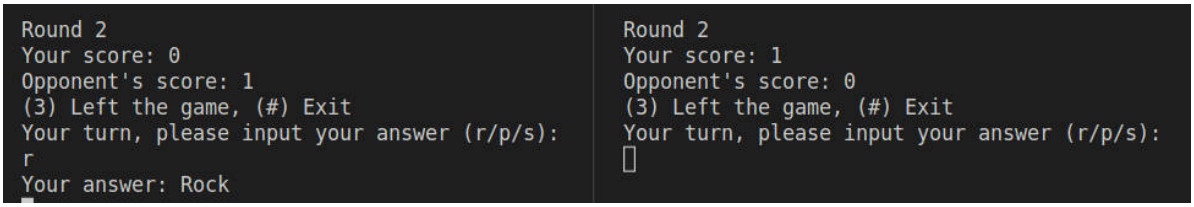
邀请好友进行游戏/接受好友的邀请

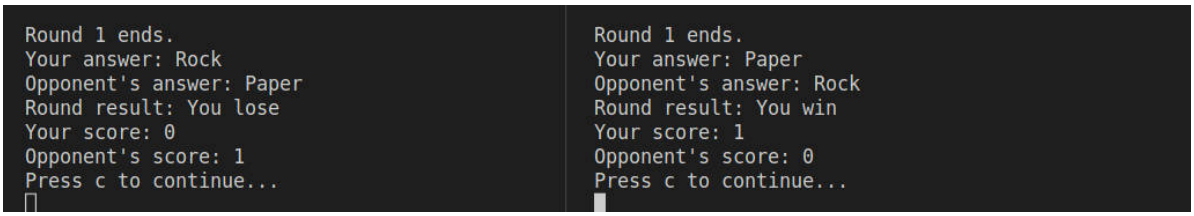
按“2”后输入好友用户名邀请好友，收到邀请后可进行同意或拒绝：



进行游戏

进行猜拳游戏，采用三局两胜制，游戏一直进行到有一方获得两分为止：





退出游戏

游戏的途中可随时退出游戏，对手也会随之退出游戏。

项目架构

服务器

服务器采用阻塞IO+select多路复用+多线程的方式实现，每个子线程服务一个客户端。没有使用更复杂的reactor模式的原因是感觉实现上更加复杂，且对于线程间同步的要求更高，出于没有大量时间学习多线程的考虑，选择最简单的子线程与客户端一一对应的模式，然而即便如此对线程间同步问题的处理几乎花费了绝大部分的项目开发时间。

每个子线程的事务处理使用循环的select调用实现，考虑到子线程没接受到数据也要进行业务处理不能一直阻塞在select调用上，使用无阻塞的IO调用循环又过于消耗cpu资源，折中选择了循环中使用加了超时值（目前设为0.5秒）的select调用的方式进行处理。在接受到数据时子线程对数据进行处理，未接收到数据时超时对共享数据进行线程间通信（同步）。可以明显减少对cpu资源的消耗。

服务器的子线程设计为无状态，因为服务器端不需要根据之前的状态来考虑下一步的操作，只需要根据客户端发送的数据（或超时）就可以决定如何处理，如接受到用户登录的数据就在用户列表中搜索等等。

客户端

客户端使用多路复用+状态机实现。根据接受到的数据或用户的输入及现在的状态决定下一步的状态，根据状态决定界面的显示。

三、实验中遇到的问题和感想

1. **pthread库提供的多线程函数过于古早和麻烦，以及缺乏线程间通信的机制，使得线程之间的同步问题非常复杂。**

   一开始使用的读写锁进行同步，然而写大半了才看到c++有新的线程库可以用，这时想改已经来不及了。至于线程间的通信问题，使用及其别扭的设置超时值的select调用，在超时后检查共享资源的更新标志值。目前看来c++没有什么好的方法实现跨线程调用，使用条件变量需要被通知的线程一直阻塞在pthread_cond_wait，然而这在一个线程服务一个客户端的架构中不可能做到，它需要一个不阻塞的事件循环，使用超时值其实是实现了一个非常的简易的定时器来实现这个循环。