基于 UDP 的即时通信程序

一、 协议设计

客户运行客户端程序，首先需要输入用户名，输入完成后客户端会向服务器发送登录请求，向服务器说明自身的用户名，IP 地址和端口，此时服务器将检测用户名是否已经被其他用户占用，若是，则回复客户端拒绝登录，客户端将重新输入用户名。若用户名可用则服务器记录客户的信息，并将其他在线的用户信息发送给客户端。之后客户端可根据来自服务器的信息选择进行通信的对方，并直接与对方进行字符通信。当客户端退出时，会先向服务器报告，服务器收到用户的退出报告后将向所有的在线用户通知退出的用户。

为了维护用户在线信息，运行中的客户端每隔一段时间必须向服务器发送一个“心跳”包来说明自己的在线状态，这个包中会包含用户的 IP 地址和端口。在服务器，每次收到一个“心跳”包都将更新用户的上次有效会话时间，并每隔一段时间进行检测，若长时间未收到用户的“心跳”包，则认为该用户意外掉线，将从在线用户中剔除该用户并通知其他用户。

协议采用多线程实现，客户端主线程与用户进行 IO 交互，另外有两个线程，一个用于定时产生“心跳”包，一个用于接收来自服务器和其他客户的消息。服务器主线程与客户端通信，另外两个线程一个用于接收来自客户端的“心跳”包并更新在线信息，另一个定时检测用户的在线状态并剔除意外掉线的用户。所有的通信都是基于 UDP 套接字。

二、 协议原语与切分

一个完整的协议原语以‘\n’字符结尾。客户端和服务器对于收到的包进行扫描，扫描到‘\n’字符则切分出原语，并输入到一个字符串流中进行原语的解析。对不合法的原语会丢弃。一个包的最后如果没有切分出完整的原语将被记录下来，与下一个收到的包的第一段切分出的原语拼接在一起作为一个完整的原语进行解析。

客户端的请求原语包括：

login <name> <ip> <port> 向服务器请求登录

bye <name> 向服务器通知离开

chat <name> <message> 向指定用户名的用户发送消息

服务器的回复原语有：

ok [<name> <ip> <port>]* 接收登录并发送在线用户信息

no 拒绝登录

offline <name> 通知用户下线

on <name> <ip> <port> 通知用户上线

三、 测试与存在的问题

程序仅在同一台主机上同时运行客户端和服务器对低并发和低请求量的情况下测试了所有功能的正常运行。由于消息接收和等待用户输入异步进行，且都采用标注 IO 流，可能在用户输入的过程中收到消息进行显示插入到未完成的用户中，但并不影响功能的使用。