服务端使用说明：

服务端运行之前需要在socket\_config.py文件中配置服务端的IP地址以及端口，然后运行server.py即可运行。

服务端的设计：

1. 数据格式

我们将程序之间的通讯划分两种消息，即系统消息“SysMsg”与聊天消息“ChatMsg”。

1. 系统消息的格式如下：

{‘SysMsg’：{a:b} },

其中a为字符串，表示系统消息的类型，b可以为字符串或者字典。

以“用户登录”为例，服务端和客户端收到的系统消息如下：

* 1. {‘SysMsg’：{“SysLoginRequest” : {username: password}}} 用于客户端发送的登录请求
  2. {‘SysMsg’：{“SysLoginAck”:“Successful Login”}} 服务端发送的登录确认
  3. {‘SysMsg’：{“SysLoginConfirmed”: username}} 客户端发送的登录确认

系统消息还包括对用户注册、查询当前在线用户、创建房间、进入房间、退出房间等请求的处理。数据格式，与上类似。

1. 聊天消息的格式如下：

{“ChatMsg”：{a:b} }

其中a为聊天消息的类型，可以为“toAll”，“toClient”以及“toRoom”，分别代表群聊消息，私聊消息，以及房间消息。

对于“toAll”类型的消息，b的格式为[sender，msg]，分别代表消息发送者的用户名以及发送的消息。

对于“toClient”以及“toRoom”类型消息，b的格式为[sender，receiver，msg]，分别代表消息发送者的用户名，接受者（用户名或房间号），以及发送的消息。

1. 数据的发送和接受。

为了解决对粘包问题的解决，程序将上述数据包加上包头，在进行发送与接收。

包头信息为“msgHeader”+ 数据包长度，其中使用struct.pack（）将数据包长度数值打包成4字节的二进制串，与“msgHeader”字符串一起构成包头，包头共13字节。

在接收端，程序首先检测包头，再对收到的数据进行读取。

1. server程序结构：

服务端使用select函数监听socket的消息，对收到的消息，按照其类型分别进行处理。

服务端同时启动两个子线程：

1. 一个线程用于处理所有客户端发送的心跳包，即“-^-^-pyHB-^-^-”，若逾期没有收到客户的心跳包，则认为该客户已意外掉线。
2. 另一个线程用于定时更新客户的在线时间。
3. 用户数据保存。

用户数据涉及到，用户名，用户密码，上次登录时间，总共在线时长，这些信息json保存在usrdata.dat文件中。

如：

{"netease1": {"totalTime": 10825.242000102997, "pwd": "1234", "lastLogin": 1518515473.689}}

其中“totalTime”记录了用户在线时长（秒），“pwd”代表了用户登录密码，“lastLogin”记录了上次登录的时间戳。