# 数据通讯程序说明文档

本程序做的是一个游戏，分为两部分，一部分是客户端，一部分是服务器。

## 游戏简介

本游戏的名字叫眼神平差大师，让玩家根据自己的经验拟合出一条直线，系统根据拟合的好坏进行排名，败者淘汰，胜者晋级。

本程序为多人游戏，允许多用户多游戏桌对战，核心玩法是服务器向每桌用户发送一系列散点，每个用户凭借自己的经验拟合出一条直线，每个用户将自己拟合出来的直线发给服务器，服务器将每人拟合的直线的残差平方和算出，根据残差平方和的大小排名，定一个名次（或积分），服务器将残差平方和、名次（积分）和样本回归线(根据散点用最小二乘法拟合的直线)算出，如果该用户拟合得好（或积分大于一阈值），则继续游戏，否则，即游戏失败，将被逐出对局，服务器将以上信息发送给客户端，客户端做出相应反应，经过多轮对决，决出获胜者。

## 类设计

服务器命名空间：LSServer

客户端命名空间：LSClient

1. 服务器

窗体类：FrmServer，用于处理服务器的界面

矩阵类：Matrix,用于处理矩阵的运算，是静态类。

桌子类：Table,表示服务器上的每张桌子。一个服务器允许有多张桌子，每张桌子上有一些座位，当桌子的位子坐满时，则可以开始游戏。

用户类：User,用于表示每个玩家，每个玩家是一个Tcpclient，与服务器的一条监听线程对应。

服务类，Service,用于处理向客户端发消息。

1. 客户端

登录窗体：FrmRoom,用于处理用户的登录

游戏窗体：FrmPlay,用于处理游戏的界面

服务类：Service,用于处理与服务器之间的通讯

通用类：General,用于放置一些类的定义。

1. 对应关系

User3

User5

User4

User2

User1

userlist

## 通讯部分

1. 服务器

本程序为TCP同步通讯。当服务器开启后，服务器将开出一个线程循环监听访问的用户（myThread,对应函数ListenClientConnect），如果监听到有客户端连接，则将这个用户加入用户表(userList),并开启一个线程(threadReceive，对应函数（ReceiveData）），监听这个客户端,如果有n个客户端接入，则服务器总共有（n+2）个线程（一个主线程，一个线性监听客户端连接，n个线程监听相应的n个客户端）。

本程序通过字符串通讯，通过设置客户端和服务器之间相应字符串之间的对应关系来进行通讯的。

每个监听线程里面有两个变量桌号:tableIndex，和座位号seat，记录该线程的玩家对应的位置。

下面将常见的字符串说明如下：

1. 登录。格式：Login,昵称。当客户端登录时，向服务器发出此消息，服务器接收到之后，将座位表字符串(格式：Table，座位字符串)发回用户。

2. 退出。格式：Logout。当客户端退出时，向服务器发出此消息，服务器收到之后，将用户从用户表和相应的桌子上清除，并退出该用户的消息循环。

3. 进入座位(坐下)。格式：SitDown, 桌号，座位号。当客户端选定位子坐下时，发此消息，服务器接收后，将相应桌号的相应座位指向该用户，表明该位子已经有人了，并向本桌其余人发送该位子的人来了的消息，向所有人发送座位表改变的消息（格式：TableChange.桌号，座位号，1，（1表示是入座））

4. 从座位上离开（站起）。格式：StandUp。当客户端从该位子站起是，发此消息，服务器接收后，将相应桌桌子的玩家移除，（但并不将该线程关闭，也并不从userlist表中去除），向所有玩家发送此消息（格式：TableChange,桌号，座位号，0）。

4. 准备游戏。格式：Ready，关数。当玩家点击准备时发送此消息，服务器将相应的玩家的准备置为true，当本桌的每个人都准备时，则发牌（格式：Deal，mode , sumPoints,x1,y1,…）

5. 画完。格式：Finish,x1,y1,x2,y2。将直线的参数或者是抛物线的参数发回，并处理，将结果发回给玩家（格式：Result,state,rank,）

6. 聊天。格式：Chat, 桌号，message。当客户端向同桌其他玩家发送消息时，会发送chat给服务器，服务器将聊天内容发给同桌其他玩家（格式：Chat，userName,message）

1. 客户端

1. 系统消息：格式：Message，msg。当其它玩家做出某些决定时，会对自己的行为有影响，将这些信息显示出来，即为系统消息。常见的有在坐下、离开时的消息。

2. 座位表：格式：Table，座位字符串，MAX\_TABLE。当玩家登录时，系统会将座位字符串发给玩家。

3. 座位表改变:格式：TableChange，桌号，座位号，1/0.当玩家进入座位或者从座位上离开时，系统将座位表改变发来。

4. 退出，格式:Logout。将退出用msgbox显示。

5. 聊天，格式：Chat.名字，内容。将谁说了什么显示在聊天框中。

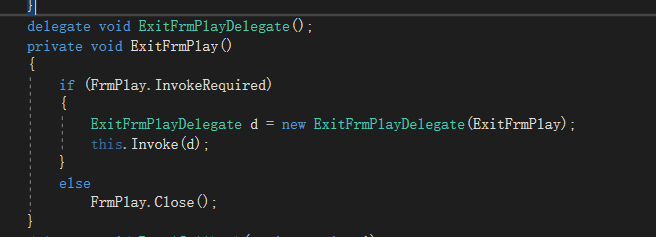
6. 断开，格式：End。服务器断开。

7. 发牌，格式：Deal，点数，坐标。用户接收到点后，将点存在数组中， 将gameStart置为true,让其绘图窗体显示散点。

## 游戏逻辑

## 细节

1. 为什么要用委托？



因为在多线程编程中，C#规定不能在别的线程调用窗体相关的语句（除非关掉这种检查），委托（C++中相当于函数指针）通过代码this.Invode(d)(其中this是指你的窗体)实现了在主线程中对该函数的调用，消除了线程安全的隐患（防止出现多线程同时修改一个窗体的情况）.

## 小组合作

本小组成员通过github进行合作及版本控制，方式是组长把程序推到github上，小组成员从组长的库中fork到自己的库中，更改后向组长提交(pull request),组长将该提交合并（merge）.

Git版本管理中，将远程库叫origin,将本地库叫master（主分支），其他分支还没有用到这里不讲。

组长master

Fetch

Push

组员master

Fetch

Push

Pull request,

merge

fork

下面针对本地库作讲解。在本地库中，将你选择建立git的文件的内容叫工作区，add之后到暂存区，commit之后到master。（主分支）

master

stage

Work directory

add

checkout

commit

reset

改乱了，回退：revet，（相当于一次提交）

在desktop中commit to master,合二为一