****

网络编程课程设计

实 验 报 告

|  |  |
| --- | --- |
| **学 院** | 网络空间安全学院 |
| **专 业** | 网络工程 |
| **班 级** | 17272412 |
| **学 号** | 17272231 |
| **学生姓名** | 吴挺锋 |
| **教师姓名** | 胡伟通 |
| **完成日期** | 2019/11/21 |
| **成 绩** |  |

**实验三 基于TCP的C/S程序设计**

1. **实验目的**
2. 熟悉TCP协议的原理和特点
3. 掌握基于TCP协议的应用层协议设计方法。
4. 掌握基于TCP协议的C/S网络应用程序设计和实现方法。
5. **实验内容**

**以下题目任选其一，或在征求指导老师同意的情况下，完成难度相似的基于TCP的网络应用。**

（1）交互式游戏设计(RemoteBet) （基于TCP）

通过在远端主机上搭建一个远程骰宝服务器，其它主机可以通过客户端程序RemoteBet与远程的骰宝服务器联系，进行交互式游戏设计。命令行格式如下：

RemoteBet <ServerIP>

如： RemoteBet 127.0.0.1

**规则如下：**

ya tc <数量> <coin|silver|gold> 押头彩(两数顺序及点数均正确) 一赔三十五

ya dc <数量> <coin|silver|gold> 押大彩(两数点数正确) 一赔十七

ya kp <数量> <coin|silver|gold> 押空盘(两数不同且均为偶数) 一赔五

ya qx <数量> <coin|silver|gold> 押七星(两数之和为七) 一赔五

ya dd <数量> <coin|silver|gold> 押单对(两数均为奇数) 一赔三

ya sx <数量> <coin|silver|gold> 押散星(两数之和为三、五、九、十一) 一赔二

每盘按从上到下的顺序**只出现一种点型**(头彩和大彩可同时出现)，其他情况都算庄家赢。

庄家唱道：新开盘！预叫头彩！

庄家将两枚玉骰往银盘中一撒。

┌───┐ ┌───┐

│● ● │ │ │

│● ● │ │ ● │

│● ● │ │ │

└───┘ └───┘

庄家唱道：头彩骰号是六、一！

输入你压的值：

ya tc 10 gold

庄家将两枚玉骰扔进两个金盅，一手持一盅摇将起来。

庄家将左手的金盅倒扣在银盘上，玉骰滚了出来。

┌───┐

│● ●│

│ ● │

│● ●│

└───┘

庄家将右手的金盅倒扣在银盘上，玉骰滚了出来。

┌───┐

│ ● │

│ │

│ ● │

└───┘

庄家叫道：五、二……七星。

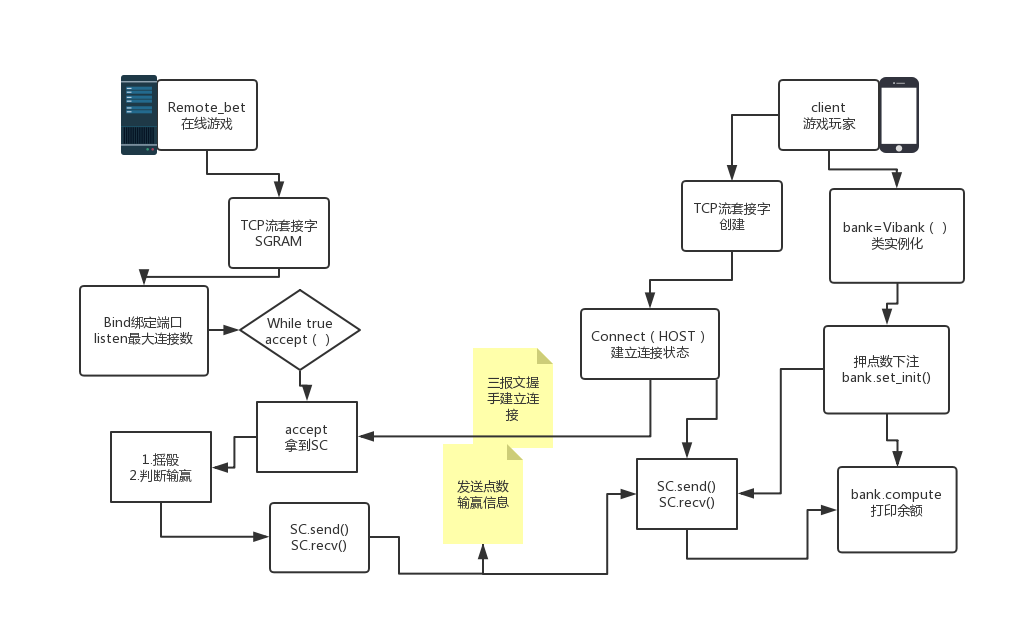
你赢了多少？or 你输了多少？

假设服务器的IP地址为127.0.0.1，客户端程序连接到服务器进行远程骰宝游戏，然后按照**服务器发回的游戏规则和提示进**行游戏。

1. **程序设计思路**
2. **所选题目说明：**

基于tcp的远程骰宝游戏。

1. 服务端充当摇骰和判断输赢的角色。故服务器会实现随机摇骰，和判断输赢的功能。将服务器里的socket创建，请求处理等封装成一个TCP\_SEVER类
2. 客户端本地则是保存着游戏的基本规则，就是投注的类型（mode），货币的等值转换（currency），各类型的赔率（rate）等。因此账户的余额加减操作，只需要服务器发送简单的判断信息，我们便可以在本地计算出来。
3. 我将会在客户端程序准备一个点数对应图像的字典，和封装一个微金库类vibank。微金库类相当于是一个小银行，里面存有汇率表，加入计算余额的功能。将它实例化，用于计算每次参与游戏后客户的余额。
4. **网络应用拓扑结构**



1. **应用层协议设计**

①服务器accept，在拿到客户端的套接字后，放到Thread类进行处理请求，开启新的线程

②服务器连续发送两个int型（点数），（传输要转成bytes）给客户端

③客户端发送ya num type coin 信息给服务器，类型是字符串转bytes

④服务器阻塞等待客户端的猜点，然后进行②步骤

⑤与此同时，服务器会算出胜利点型，并匹配，将结果传回给客户端

1. **所选用的Python库介绍**

①Threading库，用于创建多个线程处理并发性

这次用到的Thread类是threading模块的主要主要执行对象。Thread对象有三个数据属性，name(线程名)、ident(线程的标识)、daemon(布尔值，是否是守护线程)。这三个数据属性可以直接通过对象进行调用并进行设置。

②Struct库，控制数据的固定长度，可将不同类型的数据打包在一起在网络上传输。

③random库，用于生成随机数，也就是摇骰的操作。

1. **程序源代码**

**注意源代码要有详细的注释。 同学们提交的每个程序都应该遵循Honor Code（诚实代码保证）的要求。**

**请大家特别注意一定要在每个程序首部的注释中加上以下保证：**

**# 我真诚地保证：**

**# 我自己独立地完成了整个程序从分析、设计到编码的所有工作。**

**# 如果在上述过程中，我遇到了什么困难而求教于人，那么，我将在程序实习报告中**

**# 详细地列举我所遇到的问题，以及别人给我的提示。**

**# 在此，我感谢 XXX, …, XXX对我的启发和帮助。下面的报告中，我还会具体地提到**

**# 他们在各个方法对我的帮助。**

**# 我的程序里中凡是引用到其他程序或文档之处，**

**# 例如教材、课堂笔记、网上的源代码以及其他参考书上的代码段,**

**# 我都已经在程序的注释里很清楚地注明了引用的出处。**

**# 我从未没抄袭过别人的程序，也没有盗用别人的程序，**

**# 不管是修改式的抄袭还是原封不动的抄袭。**

**# 我编写这个程序，从来没有想过要去破坏或妨碍其他计算机系统的正常运转。**

**# <学生姓名> 吴挺锋**

1. **服务器端源码**

from socket import \*

import random,sys

import struct,threading

*#1.判断奇偶*

**def** oe(s):

    if (s%2) == 0:

        return 0

    else:

        return 1

*#2.扔骰子*

**def** random\_dice():

    n = random.randint(1,6)*#随机返回1-6范围内的一个整数，包括边界*

    return str(n)

*#3.循环发骰子,返回点数记录的列表*

**def** roll\_send(sc,len):

    a=[]

    for m in range(len):

        m = random\_dice()

        sc.sendall(m.encode('utf-8'))    *#Tcp sendall*

        a.append(m)

    print(a)

    return a

*#4.生成这局赢家type*

**def** judge\_dice(n1,n2,an1,an2):     *#该局结果*

    n1=int(n1)

    n2=int(n2)

    an1=int(an1)

    an2=int(an2)

    if (n1==an1) and (n2==an2): *#1.头彩，第一个对于第一个，第二个对应第二个*

        return 'tc'

    elif ((n1==an1)and(n2==an2))or((n1==an2)and(n2==an1)):*#2.大彩顺序可颠倒*

        return 'dc'

    elif (an1!=an2) and (oe(an1)==0) and (oe(an2)==0):*#3.空盘，自己的an1，不一样*

        return 'kp'

    elif ((an1+an2)==7):*#4.七星，既加起来是7*

        return 'qx'

    elif (oe(an1)==1) and (oe(an2)==1):*#5.单对，两奇数*

        return 'dd'

    elif ((an1+an2)==3)or((an1+an2)==5)or((an1+an2)==9)or((an1+an2)==11):*#6.散星*

        return 'sx'

    else:*#从服务端，这种情况不存在*

        return '胜利点型不存在！'

        sys.exit()

**class** TCP\_SERVER:

**global** tcp\_server

**def** server\_init(self):

        server\_addr = ('127.0.0.1',8880)

        server = socket(AF\_INET,SOCK\_STREAM)   *#区别UDP是Stream流传输,创建套接字*

        server.bind(server\_addr)       *#绑定地址（端口号）*

        server.setsockopt(SOL\_SOCKET,SO\_REUSEADDR,1) *#设置端口复用*

        server.listen(64)

        return server

**def** handle\_request(self,sc,sockname):

        print('{} 加入了游戏'.format(sockname))        *#记录玩家信息*

        while True:

            Fir=roll\_send(sc,2)*#庄家先叫两个,返回记录*

            data = sc.recv(256).decode('utf-8')  *#玩家要叫骰子，所以要接受数据*

            Sec=roll\_send(sc,2)

            if not data or data == 'q':

                sc.close()*#关闭套接字*

                print('{} 已退出了游戏'.format(sockname))

                sys.exit(0)

            print('{} 玩家 {}说：'.format(sockname,data))  *#玩家结果*

            yaType=data.split(' ')[1]

*#把列表拼起来 ,,,,算出这局赢的点型*

            r=Fir+Sec

*#只有你压的类型和这个相等就OK,then发送定长数据*

            win\_type=judge\_dice(r[0],r[1],r[2],r[3])

            if yaType==win\_type:

                sc.sendall(struct.pack('4s2s',**b**'win',win\_type.encode('utf-8')))

            else:

                sc.sendall(struct.pack('4s2s',**b**'lose',win\_type.encode('utf-8')))

            print('{}已发送')

*##这里要阻塞一下*

            s=sc.recv(10)

            print('玩家 {} 开始下一次'.format(sockname))

**def** server\_run(self):

**global** tcp\_server

        tcp\_server=self.server\_init()

        print('远程骰庄游戏 {}'.format('127.0.0.1'))

        while True:

            sc,ip\_port=tcp\_server.accept()*#接收请求*

            print("[新玩家]:", ip\_port, "已加入")

*# 有客户端连接后，创建一个线程将客户端套接字，IP端口传入recv函数，*

            try:

                t1 = threading.Thread(target=self.handle\_request, args=(sc, ip\_port))

*# 设置线程守护*

                t1.setDaemon(True)

*# 启动线程*

                t1.start()

            except Exception:

                print('catch that')

**def** close(self):

        self.tcp\_server.close()

if \_\_name\_\_ == '\_\_main\_\_':

    try:

        tcp\_sc=TCP\_SERVER()

        tcp\_sc.server\_run()

    except KeyboardInterrupt:

        tcp\_sc.close()

        sys.exit(0)

1. **客户端源码**

from socket import \*

import sys,struct

SMALL=1

*#1.建立tcp套接字,连接server端*

Server\_addr = ('127.0.0.1',8880)

client = socket(AF\_INET,SOCK\_STREAM)

client.connect(Server\_addr)

*#1.规则描述*

dd= {'tc': '押头彩:两束顺序点数均正确xx1:35', 'dc': '押大彩:两束点数正确xx1:17',

'kp' : '押空盘:两数不同且均为偶数xx1:5', 'qx' :'押七星:两数之和为七xx1:5',

'dd': '押单对:两数均为奇数xx1:3' ,'sx' : '押散星:两数之和为三、五、九、十一xx1:2'}

*#2.赔率表声明*

keys=[]

rates=[]

*#1-1.规则介绍*

print('-------------------------------------------')

print('欢迎来到Remote Bet,规则如下：')

for key in dd.keys():*#遍历key就好*

    val=dd[key].split('xx')

    print('ya {} <数量> <coin|silver|gold> {**:<20s**} {**:6s**}'.format(key,val[0],val[1]))

    keys.append(key)

    rates.append(val[1].split(':')[1])

*#2-1.赔率表出来了*

dict\_rates=dict(zip(keys,rates))

print(dict\_rates)

*#骰子图样字典 三个单引号*

dict1 = {'1':'''

          ┌───┐

          │   │

          │ ● |

          │   │

          └───┘

          ''',

         '2':''''

          ┌───┐

          │ ● │

          │   │

          │ ● │

          └───┘

          ''',

         '3':'''

          ┌───┐

          │ ● │

          │   │

          │● ●│

          └───┘

          ''',

         '4':'''

          ┌───┐

          │● ●│

          │   │

          │● ●│

          └───┘

          ''',

         '5':'''

          ┌───┐

          │● ●│

          │ ● │

          │● ●│

          └───┘

          ''',

         '6':'''

          ┌───┐

          │● ●│

          │● ●│

          │● ●│

          └───┘

          '''

                }

dict2 = {'1':'一','2':'二','3':'三','4':'四','5':'五','6':'六'}       *#骰子点数字典*

*#bytes-->utf-8*

**def** b2u (s):

    return s.decode('utf-8')

*#个人微金库类创建*

**class** vibank:

*#1.汇率比例*

    coin = 1

    silver = 10 \* coin

    gold = 100 \* coin

    balance = 100

*#2.记录押注金额和类型*

    bet = coin *#default,初始一coin*

    mode = '' *#tc,dc,dd,sd,sx*

    csg='' *#coin silver,gold*

*#重新设置*

**def** set\_init(self,typ,bet,csg):

        self.bet=int(bet)

        self.mode=typ*#*

        self.csg=csg

*#4.计算余额,下注类型出来*

**def** compute(self,flag):

**global** dict\_rates

        rate=int(dict\_rates[self.mode])*#赔率转成整型*

        if self.csg=='coin':

            er=self.coin

        elif self.csg=='silver':

            er=self.silver

        else:

            er=self.gold

*#货币单位调整完毕，押数x汇率x赔率*

        if flag!='lose':

            self.balance+=self.bet\*er\*rate

            print('您押对了，恭喜')

        else:

            self.balance-=self.bet\*er\*rate

            print('您压错了，别灰心')

        print('您的余额是:{}'.format(self.balance))

        self.bet=1  *#default=1*

        return self.balance

cus=vibank()

*#循环是为了保证能持续进行通话*

while True:

    print('''

        庄家唱道：新开盘！预叫头彩！

        庄家将两枚玉骰往银盘中一撒。

        ''')

**global** n1,n2,an1,ya

    n1 = b2u(client.recv(SMALL))   *# 因为1个数字只占一个字节，接受服务器端发来的点数*

    n2 = b2u(client.recv(SMALL))

    print(dict1[n1],dict1[n2])         *#输出对应点数的预备字典*

    print('头彩骰号是',dict2[n1],dict2[n2])

    data = input('输入你押的值  (ya <玩法> <数量> <coin|silver|gold>) ：')*#ya dd 2 coin*

*##分析data*

    if (data != 'q'):

        client.sendall(data.encode('utf-8'))

*#分离data数据，tyoe num mode初始化*

        li=data.split(' ')

        cus.set\_init(li[1],li[2],li[3])

    else:

        client.send(data.encode('utf-8'))*#data='exit'*

        sys.exit()

*##准备接收点数*

    print("""

        庄家将两枚玉骰扔进两个金盅，一手持一盅摇将起来。

        庄家将左手的金盅倒扣在银盘上，玉骰滚了出来。""")

    an1 = b2u(client.recv(SMALL))

    print(dict1[an1])

    print("""

        庄家将右手的金盅倒扣在银盘上，玉骰滚了出来。""")

    bn2 = b2u(client.recv(SMALL))

    print(dict1[bn2])

*#接受，庄家判输赢*

    result=client.recv(6)

    res,win\_type=struct.unpack('4s2s',result)

    print('庄家叫道：{}、{}'.format(dict2[an1],dict2[bn2]))

    print('庄家公布赢的点型 {},你的是{}'.format(b2u(win\_type),cus.mode))

    balance=cus.compute(b2u(res))

    s=client.sendall(**b**'ok')

1. **程序测试方法及测试结果记录（不能光截图，要有相应的文字说明）**
2. **测试方法**

①先打开服务器端

②打开一个客户端进行交互，进行押骰判骰的操作。

③打开两个客户端进行交互。

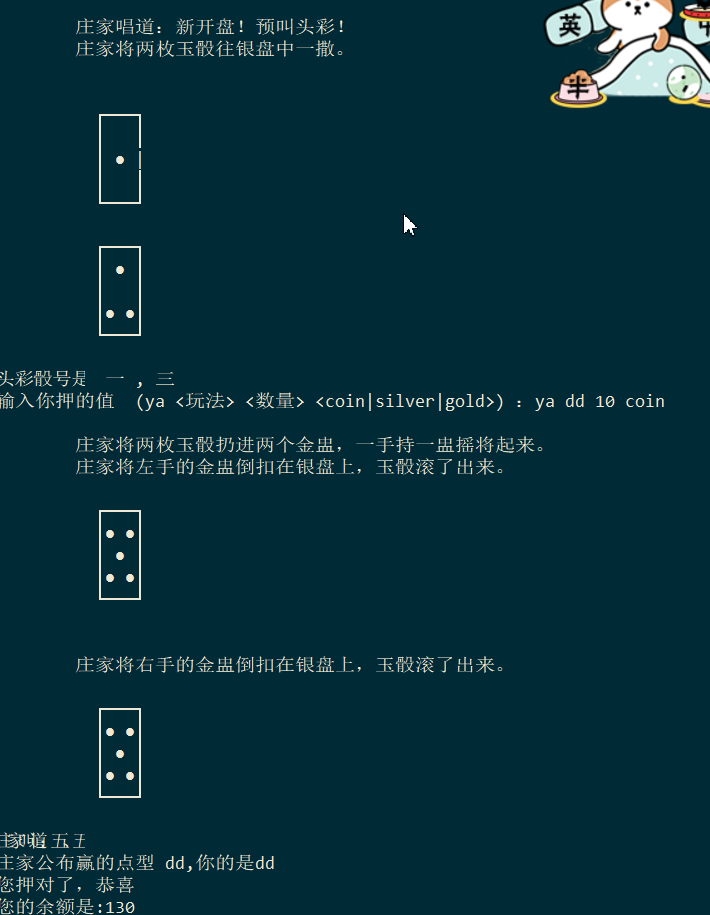
1. **测试流程**

①先打开服务端，在打开客户端

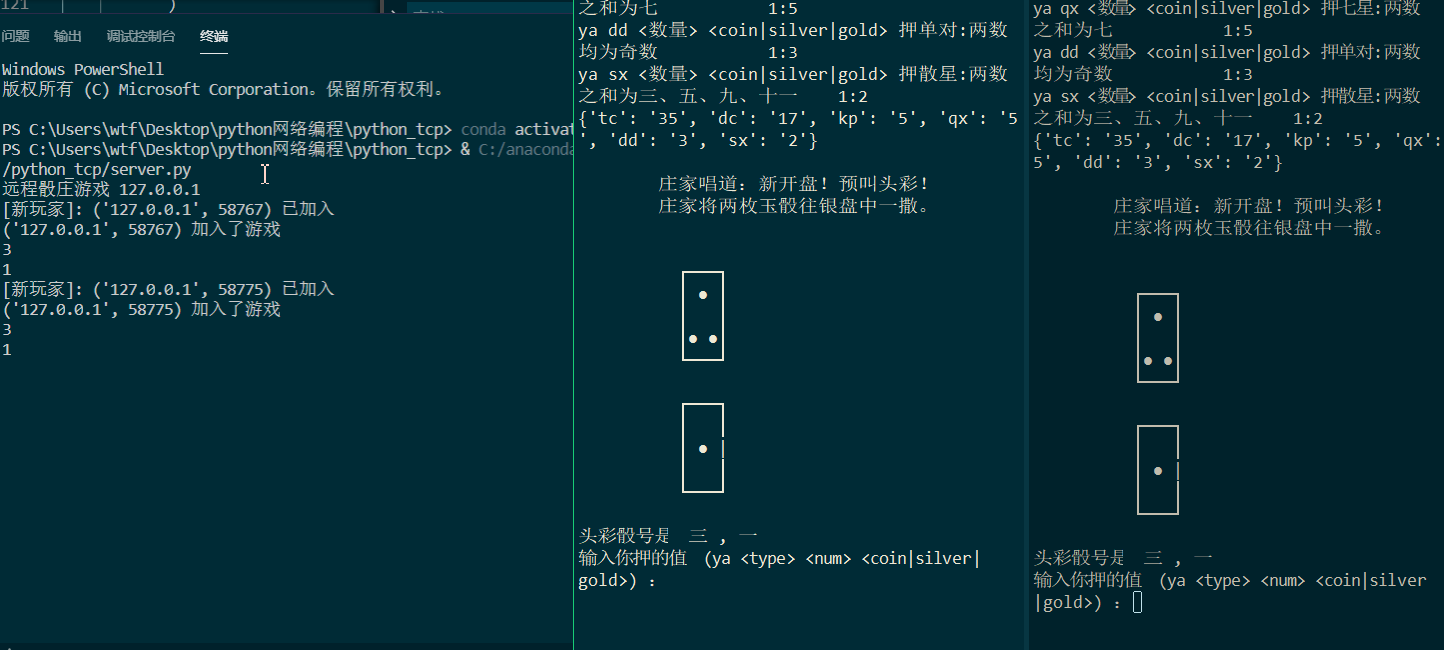


可以按提示进行押骰操作

比如我下面这里押 两数和为奇数，正好赢了，赔率1:3，故余额100+3\*10=130元是正确的



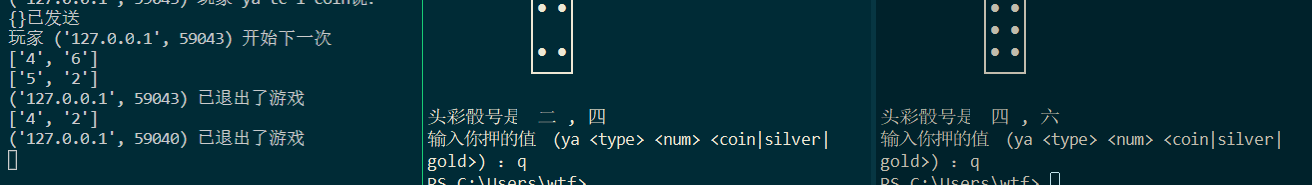
同时开启两个客户端。服务器创建新的线程来处理请求



继续跟着提示押骰



一样计算出来，最后键入‘q’正常退出。



1. **实验分析总结及心得**

（结合所学知识对实验过程中观察到的实验结果进行分析总结，以便加深对知识的理解，并总结通过实验学到的知识或技术）

①TCP套接字与之前的UDP套接字截然不同，建立在有连接的基础上。Tcp的套接字又分为主动套接字和被动套接字，起初是没有完全理解，不知道该如何选择，后面通过复习课本老师讲过的知识，正好也存在笔记，通过范例去学习。TCP客户用connect函数来建立一个与TCP服务器连接，

connect函数：sockfd是由socket函数返回的套接字描述符，第二个、第三个参数分别是指向一个套接字地址结构的指针和该结构的大小，套接字结构必须含有服务器的IP地址和端口号。而socket创建一个套接字时，它被假设为一个主动套接字，也就是说，它是一个将调用connect发起连接的一个客户套接字。

listen函数：把一个未连接的套接字转换为一个被动套接字，指示内核应接受指向该套接字的连接请求，调用listen函数将导致套接字从CLOSEE状态转换到LISTEN状态。

accept函数：TCP服务器调用，从已完成队列中列头返回下一个已完成连接，也就是拿到与客户连接的套接字。理解到这个，我就去设置了listen来构造一个等待连接的队列，通过accept拿出套接字。

②多线程Threading之thread类的使用，起初我觉得这个库十分的晦涩，因为在我上一个UDP实验的时候，也没有选择这个库，因为当时UDP+thread的范例是在太少，出来的基本都是TCP的。正好借着TCP来学习一下thread类的使用，也是因为一开始没考虑并发，也导致实验报告拖延了一个星期。后面，我找到了合适的范例。介绍了怎么把请求处理丢进thread类创造一个新的线程，于是我发现在服务器端，正好可以把这些代码块，方法，组合成一个新的server类，最后我整理了原有的代码，起初调试一直有些毛病，也比较急躁，通过设置断点调试，才逐渐找到小毛病，最后是成功啦。

③关于骰子的算法还有图案问题，我并不知道该怎么合适的处理，最后是从老师的提示当中明白要用字典来进行存储。网上资料说三个单引号包围能把换行也给显示出来，最后是采取这个方法解决的。

④关于网络异常的问题，我觉得还十分重要的。周三课上老师说到异常，才感觉恍然大悟，这不就是我遇到的问题吗。不过这块有点难学，要继续努力才行，总之还是顺利完成了，只不过拖得有点迟了。。