****

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** | 18272412 |
| **学 号** | 18041618 |
| **学生姓名** | 廖越强 |
| **教师姓名** | 胡伟通 |
| **完成日期** | 2020.11.4 |
| **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。

（2）以代码方式解析Wireshark抓包文件并绘制图形

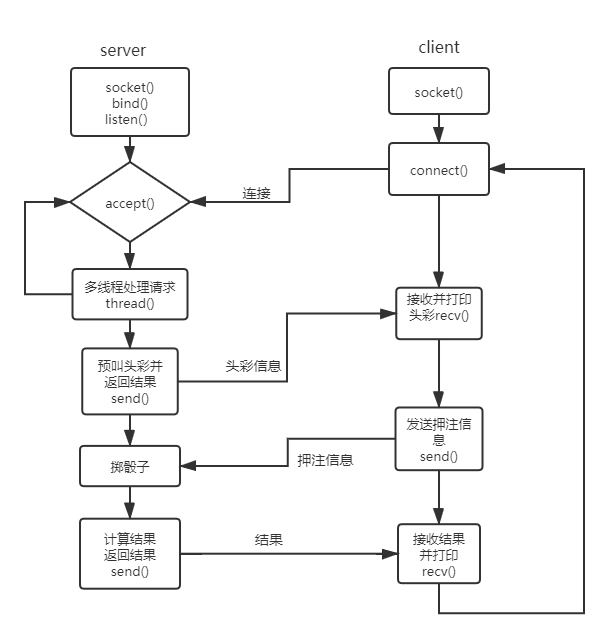
首先打开Wireshark，在联网的网卡上开始抓包过程，再用浏览器打开链接http://cbs.hdu.edu.cn/\_upload/article/images/f5/2a/c604697f45e1a01fd95461626dd0/082e8e2d-2068-4afa-b783-4cde6a7978a1.jpg，关闭网页并结束抓包过程，然后将抓包内容保存为pcapng格式的文件。

对上述pcapng文件，使用Scapy库或其他解析库，获取到tcp三报文连接或tcp四报文释放的信息，获取到相关信息后，再用Matplot库绘制三报文连接或四报文释放的过程。

1. **程序设计思路**
2. **所选题目说明：**

交互式游戏设计(RemoteBet) （基于TCP）

1. **网络应用拓扑结构**



1. **应用层协议设计**

骰子点数(2字节)：第0、1字节分别表示第一、二个骰子的点数

骰子点型(2字节)：用于表示如头彩tc，大彩dc等结果

获得金额(第5字节往后)：本轮得到的金额

押注信息(数量，蛊类型，点型)以空格分开，直接作为字符串信息发送

1. **所选用的Python库介绍**

1．random 生成伪随机数（用于生成骰子点数）

2．socket 网络编程接口

3．threading 该模块提供了操作多个线程的底层原语。

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，**

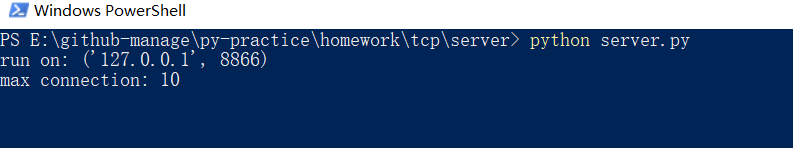
**# 不管是修改式的抄袭还是原封不动的抄袭。**

**# 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。**

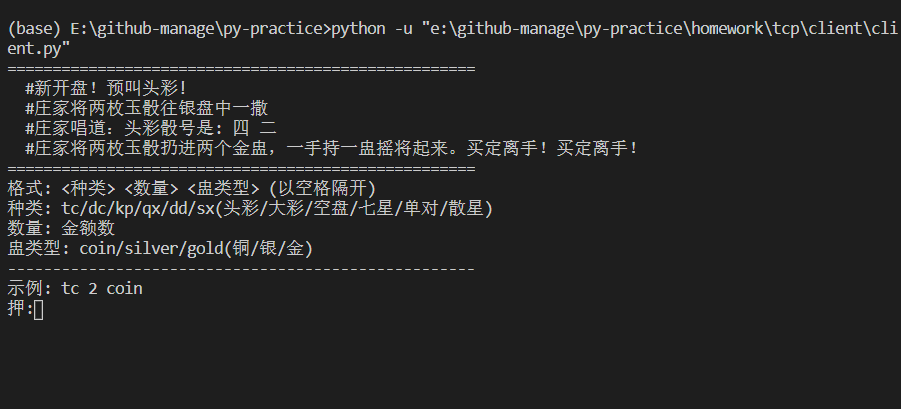
**# <学生姓名> 廖越强**

1. **服务器端源码**
2. import random
3. import socket
4. import threading
5. num\_dict = {1: "一", 2: "二", 3: "三", 4: "四", 5: "五", 6: "六"}
6. money\_dict = {'gold': '金', 'sliver': '银', 'coin': '铜'}
7. money\_num = {"gold": 50, "sliver": 50, "coin": 50}
8. def touzi\_start():
9. res = ""
10. num1 = random.randint(0, 5)
11. num2 = random.randint(0, 5)
12. res = res + str(num1)
13. res = res + str(num2)
14. print(res)
15. return res
16. def calcu(touzi, pre\_touzi, u\_opt):
17. u\_opt = u\_opt.split(' ')
18. money = ''
19. reward\_type = ''
20. res = ''  # 返回的数据: reward\_type + money
21. t1 = int(touzi[:1]) + 1
22. t2 = int(touzi[1:2]) + 1
23. print(t1, t2)
24. sum = t1 + t2
25. p\_t1 = int(pre\_touzi[:1]) + 1
26. p\_t2 = int(pre\_touzi[1:2]) + 1
27. # 判断骰子点型
28. if t1 == p\_t1 and t2 == p\_t2 or t1 == p\_t2 and t2 == p\_t1:
29. reward\_type = 'dc'
30. print('dc')
31. if touzi == pre\_touzi:
32. print('tc')
33. reward\_type = 'tc'
34. elif t1 != t2 and t1 % 2 == 0 and t2 % 2 == 0:
35. reward\_type = 'kp'
36. print('kp')
37. elif t1+t2 == 7:
38. reward\_type = 'qx'
39. print('qx')
40. elif t1 % 2 == 1 and t2 % 2 == 1:
41. reward\_type = 'dd'
42. print('dd')
43. elif sum == 3 or sum == 5 or sum == 9 or sum == 11:
44. reward\_type = 'sx'
45. print('sx')
46. else:
47. reward\_type = 'zy'
48. # 计算奖励
49. if u\_opt[0] == 'tc':
50. if touzi == pre\_touzi:
51. money = str(int(u\_opt[1]) \* money\_num[u\_opt[2]] \* 35)
52. else:
53. money = '0'
54. elif u\_opt[0] == 'dc':
55. if t1 == p\_t1 and t2 == p\_t2 or t1 == p\_t2 and t2 == p\_t1:
56. money = str(int(u\_opt[1]) \* money\_num[u\_opt[2]] \* 17)
57. else:
58. money = '0'
59. elif u\_opt[0] == 'kp':
60. if t1 != t2 and t1 % 2 == 0 and t2 % 2 == 0:
61. money = str(int(u\_opt[1]) \* money\_num[u\_opt[2]] \* 5)
62. else:
63. money = '0'
64. elif u\_opt[0] == 'qx':
65. if t1+t2 == 7:
66. money = str(int(u\_opt[1]) \* money\_num[u\_opt[2]] \* 5)
67. else:
68. money = '0'
69. elif u\_opt[0] == 'dd':
70. if t1 % 2 == 1 and t2 % 2 == 1:
71. money = str(int(u\_opt[1]) \* money\_num[u\_opt[2]] \* 3)
72. else:
73. money = '0'
74. elif u\_opt[0] == 'sx':
75. if sum == 3 or sum == 5 or sum == 9 or sum == 11:
76. money = str(int(u\_opt[1]) \* money\_num[u\_opt[2]] \* 2)
77. else:
78. money = '0'
79. else:
80. money = "invalid\_input"
81. res = reward\_type + money
82. return res
83. def game\_start(client\_socket):
84. pre\_touzi = touzi\_start() # 预叫头彩!
85. client\_socket.send(pre\_touzi.encode("utf-8"))
86. recv\_data = client\_socket.recv(1024)  # 玩家押注
87. print("玩家押注: %s" % recv\_data.decode("utf-8"))
88. touzi = touzi\_start() # 正式摇筛子
89. res = calcu(touzi, pre\_touzi, recv\_data.decode("utf-8"))
91. client\_socket.send((touzi+res).encode("utf-8"))  # 把骰子的结果(2位)和押注结果( reward\_type(2位) + money )拼接后返回
92. def main():
93. addr = ("127.0.0.1", 8866)
94. # 创建套接字
95. tcp\_server\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)
96. # 绑定IP和Port
97. tcp\_server\_socket.bind(addr)
98. print("run on:", addr)
99. # 限制连接数量
100. tcp\_server\_socket.listen(10)
101. print("max connection: 10")
102. while(True):
103. # 等待客户端的连接, accept返回(conn, address)其中conn是一个新的套接字对象，用于在此连接上收发数据
104. client\_socket, client\_addr = tcp\_server\_socket.accept()
105. print("connect from:", client\_addr)
106. # 多线程处理
107. my\_thread = threading.Thread(target=game\_start, args=(client\_socket,))
108. my\_thread.start()
109. # 关闭套接字
110. client\_socket.close()
111. tcp\_server\_socket.close()
112. if \_\_name\_\_ == "\_\_main\_\_":
113. main()
114. **客户端源码**
115. import socket
116. num\_dict = {0: "一", 1: "二", 2: "三", 3: "四", 4: "五", 5: "六"}
117. tz\_type\_dict = {
118. 'tc': '头彩',
119. 'dc': '大彩',
120. 'kp': '空盘',
121. 'qx': '七星',
122. 'dd': '单对',
123. 'sx': '散星',
124. 'zy': '庄赢'
125. }
126. touzi = [
127. """
128. ┌─────┐
129. │     │
130. │  ●  │
131. │     │
132. └─────┘""", """
133. ┌─────┐
134. │  ●  │
135. │     │
136. │  ●  │
137. └─────┘""", """
138. ┌─────┐
139. │  ●  │
140. │     │
141. │ ● ● │
142. └─────┘""", """
143. ┌─────┐
144. │ ● ● │
145. │     │
146. │ ● ● │
147. └─────┘""", """
148. ┌─────┐
149. │ ● ● │
150. │  ●  │
151. │ ● ● │
152. └─────┘""", """
153. ┌─────┐
154. │ ● ● │
155. │ ● ● │
156. │ ● ● │
157. └─────┘"""
158. ]
159. def res\_print(res):
160. print("res", res)
161. print("庄家将左手的金盅倒扣在银盘上，玉骰滚了出来。")
162. print(touzi[int(res[:1])])
163. print("庄家将右手的金盅倒扣在银盘上，玉骰滚了出来。")
164. print(touzi[int(res[1:2])])
165. print("庄家叫道:", tz\_type\_dict[res[2:4]])
166. print("本次收入:", res[4:])
167. def main():
168. dest\_ip = '127.0.0.1'
169. dest\_port = 8866
170. while True:
171. tcp\_socket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) # 创建套接字socket
172. dest\_addr = (dest\_ip, dest\_port)
173. tcp\_socket.connect(dest\_addr)
174. recv\_data = tcp\_socket.recv(1024)  # 预叫头彩信息
175. t1 = int(recv\_data.decode("utf-8")[:1])
176. t2 = int(recv\_data.decode("utf-8")[1:2])
177. print("====================================================")
178. print("  #新开盘！预叫头彩!")
179. print("  #庄家将两枚玉骰往银盘中一撒")
180. print("  #庄家唱道：头彩骰号是:", num\_dict[t1], num\_dict[t2])
181. print("  #庄家将两枚玉骰扔进两个金盅，一手持一盅摇将起来。买定离手！买定离手！")
182. print("====================================================")
183. print("格式: <种类> <数量> <盅类型> (以空格隔开)")
184. print("种类: tc/dc/kp/qx/dd/sx(头彩/大彩/空盘/七星/单对/散星)")
185. print("数量: 金额数")
186. print("盅类型: coin/silver/gold(铜/银/金)")
187. print("----------------------------------------------------")
188. print("示例: tc 2 coin")
189. send\_data = input("押:")
190. tcp\_socket.send(send\_data.encode("utf-8")) # 发送玩家押注信息
191. recv\_data = tcp\_socket.recv(1024)  # 接收押注结果
192. res\_print(recv\_data.decode("utf-8"))
193. print('\n')
194. tcp\_socket.close()
195. if \_\_name\_\_ == "\_\_main\_\_":
196. main()
197. **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）**
198. **测试方法**
199. 启动服务器，等待连接
200. 启动客户端，连接服务器
201. 亲自试玩，检验逻辑合理性
202. **测试流程**

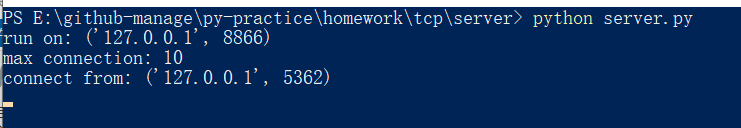
服务端启动



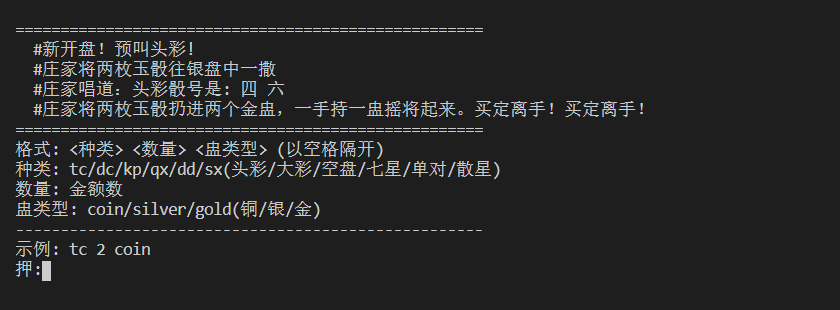
启动客户端，连接服务器后得到预叫头彩(骰子的点数)



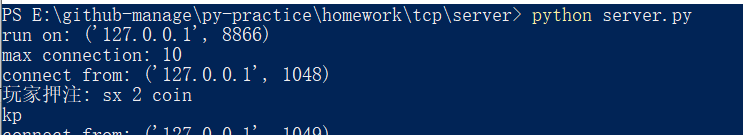
服务端收到请求，打印客户端的连接信息



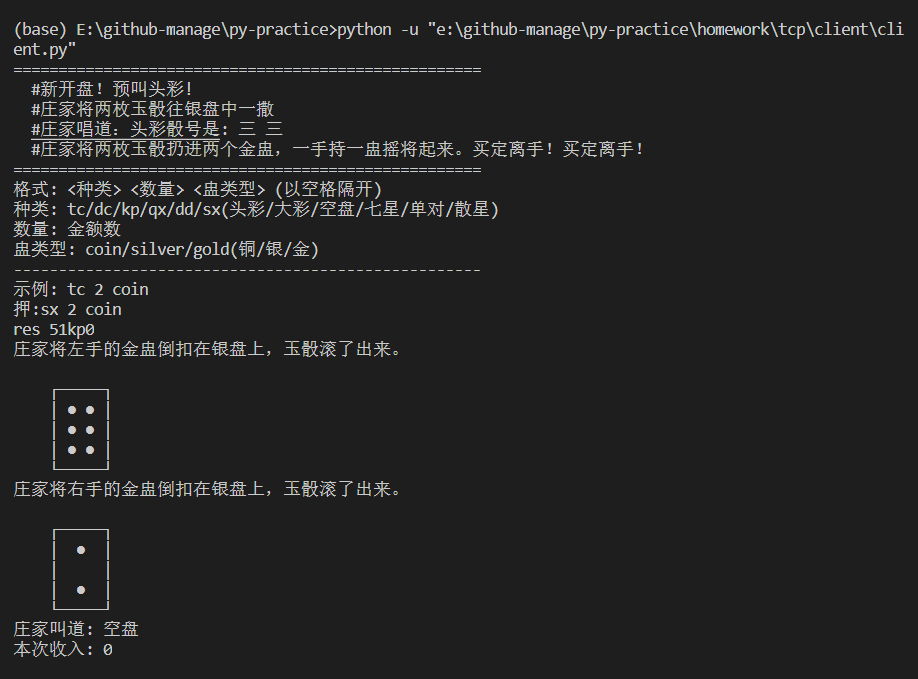
玩家输入押注信息 sx 2 coin（散星/数量2/铜蛊）



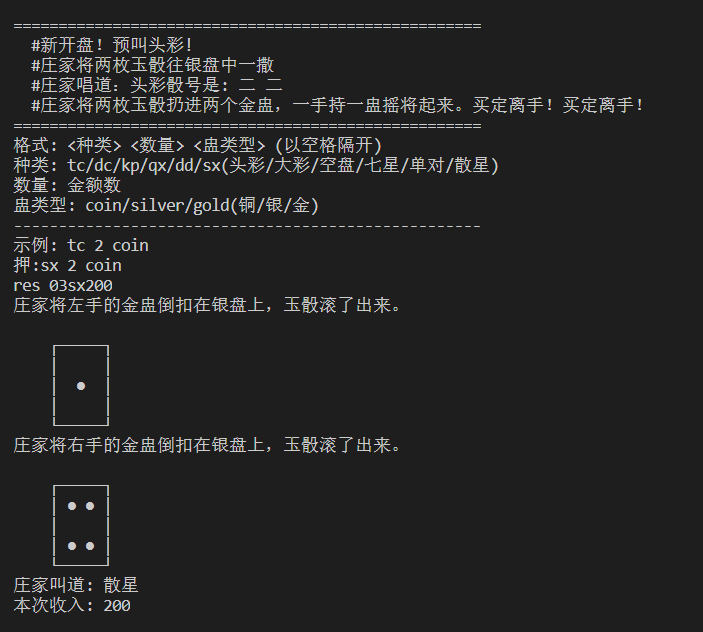
服务端接收押注信息，计算结果后返回给客户端



客户端收到结果，实际点数为六，二，点型为空盘，这次就没赚到钱。



经过多次尝试，终于赢了一次。押注的点型是散星(一赔二)，数量为2，coin每份为50，因此得到的奖励是2 \* 50 \* 2 = 200！



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）

特别推荐学生写出做实验时遇到的问题，并描述从原理分析得到解决方案的过程。

本次实验跟上一次的UDP还是挺相似的，因此做起来也非常地熟悉顺手，没有遇到特别奇怪的问题。使用socket创建TCP连接与UDP的连接的差别，大概就是TCP这边server端多了一个listen来限制连接数，而client端则是需要先进行connect()来实现连接。

在最初实现了基本功能后（单个玩家能玩），考虑到可能会有其他玩家同时也想玩，于是又在server端处理请求的方式做了些修改，引入了threading模块，在接收到一个玩家连接请求后，启动一个线程。