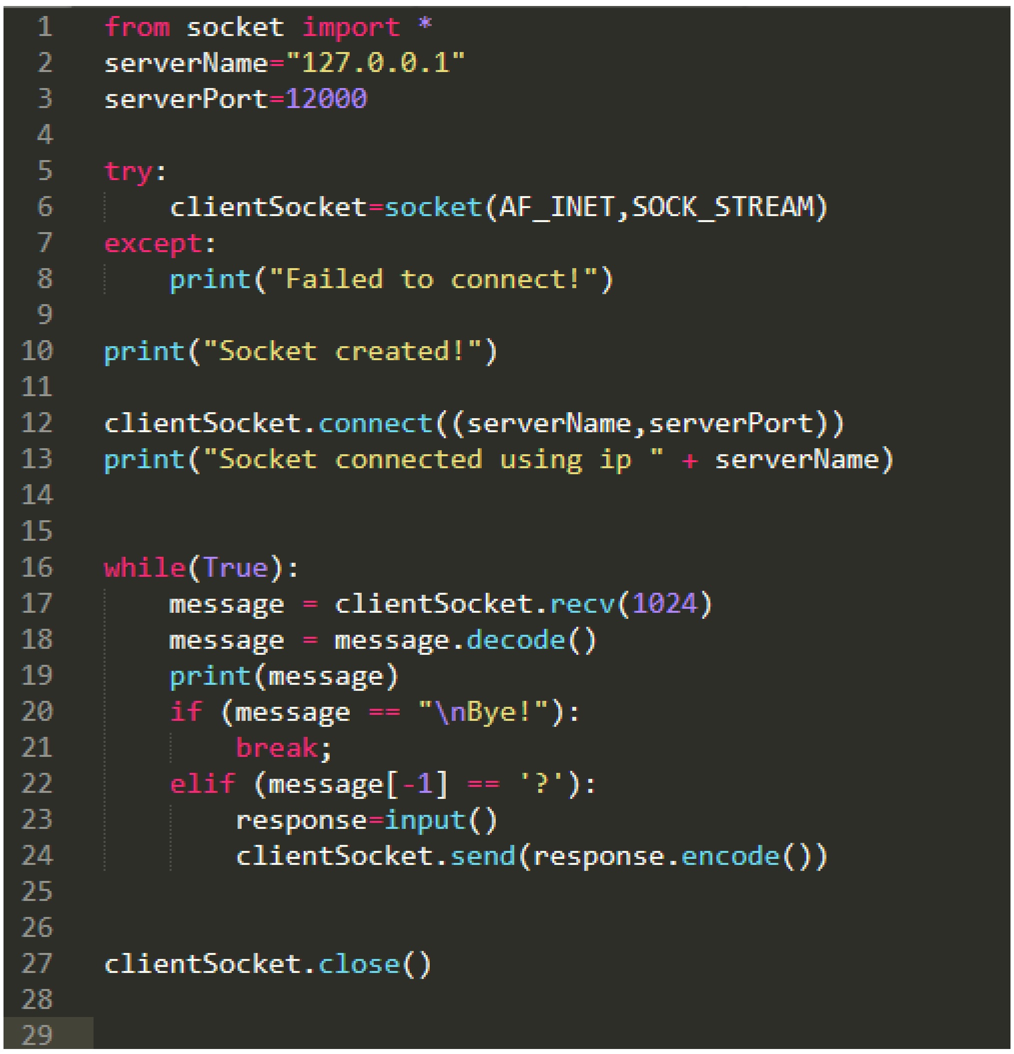
BLG 433E COMPUTER COMMUNICATIONS
2019-2020 FALL / PROJECT 1 REPORT
SOCKET PROGRAMMING

Kubilay Yazoğlu - 150140034

Screenshots from the game are added to the end of the report.

1) Code Files

a) Client.py

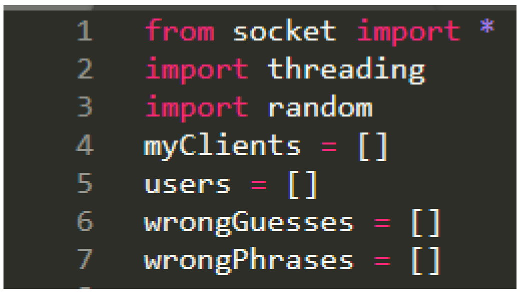


```
1   from socket import *
2   serverName="127.0.0.1"
3   serverPort=12000
4
5   try:
6       clientSocket=socket(AF_INET,SOCK_STREAM)
7   except:
8       print("Failed to connect!")
9
10  print("Socket created!")
11
12  clientSocket.connect((serverName,serverPort))
13  print("Socket connected using ip " + serverName)
14
15
16  while(True):
17      message = clientSocket.recv(1024)
18      message = message.decode()
19      print(message)
20      if (message == "\nBye!"):
21          break;
22      elif (message[-1] == '?'):
23          response=input()
24          clientSocket.send(response.encode())
25
26
27  clientSocket.close()
28
29
```

Client.py file is pretty basic. Between 5th and 13th lines, connection to server socket using TCP is done. At the 16th line, client starts listening the server. If client gets a message, it prints out to its screen. If the message is “Bye!” then client stops listening server and closes the connection afterwards. This is done if client doesn’t want to play anymore. We’ll see more about this in server.py. If the incoming message has a “?” in the end, that means client is supposed to answer that message. So we wait for a response at 23th line. When there is a response, client sends it to the server.

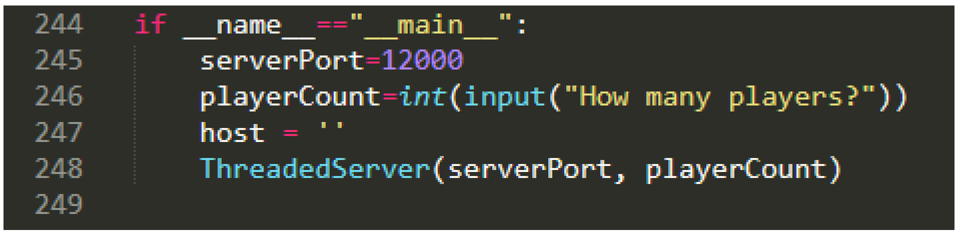
b) server.py

Before we move into the functions, let’s see our imports and global variables.



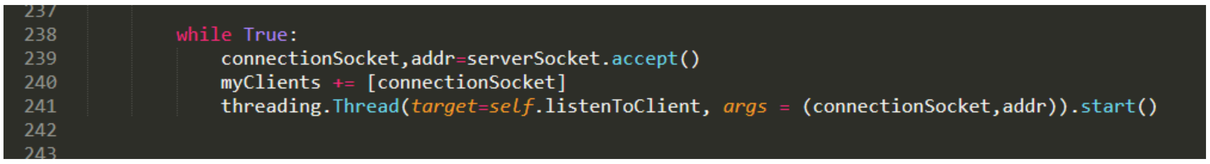
```
from socket import *
import threading
import random
myClients = []
users = []
wrongGuesses = []
wrongPhrases = []
```

Socket and threading are imported to manage connections. Random is imported to select a random word from word.txt file. Other 4 lists are initialized empty. They will be used in game to keep track of the states.



```
244  if __name__=="__main__":
245      serverPort=12000
246      playerCount=int(input("How many players?"))
247      host = ''
248      ThreadedServer(serverPort, playerCount)
249
```

Program starts from 244th line. Port and host are given here. Number of players needed to start the game is asked to server owner. Afterwards, server is initiated using these three information in the init function of the Threadedserver.



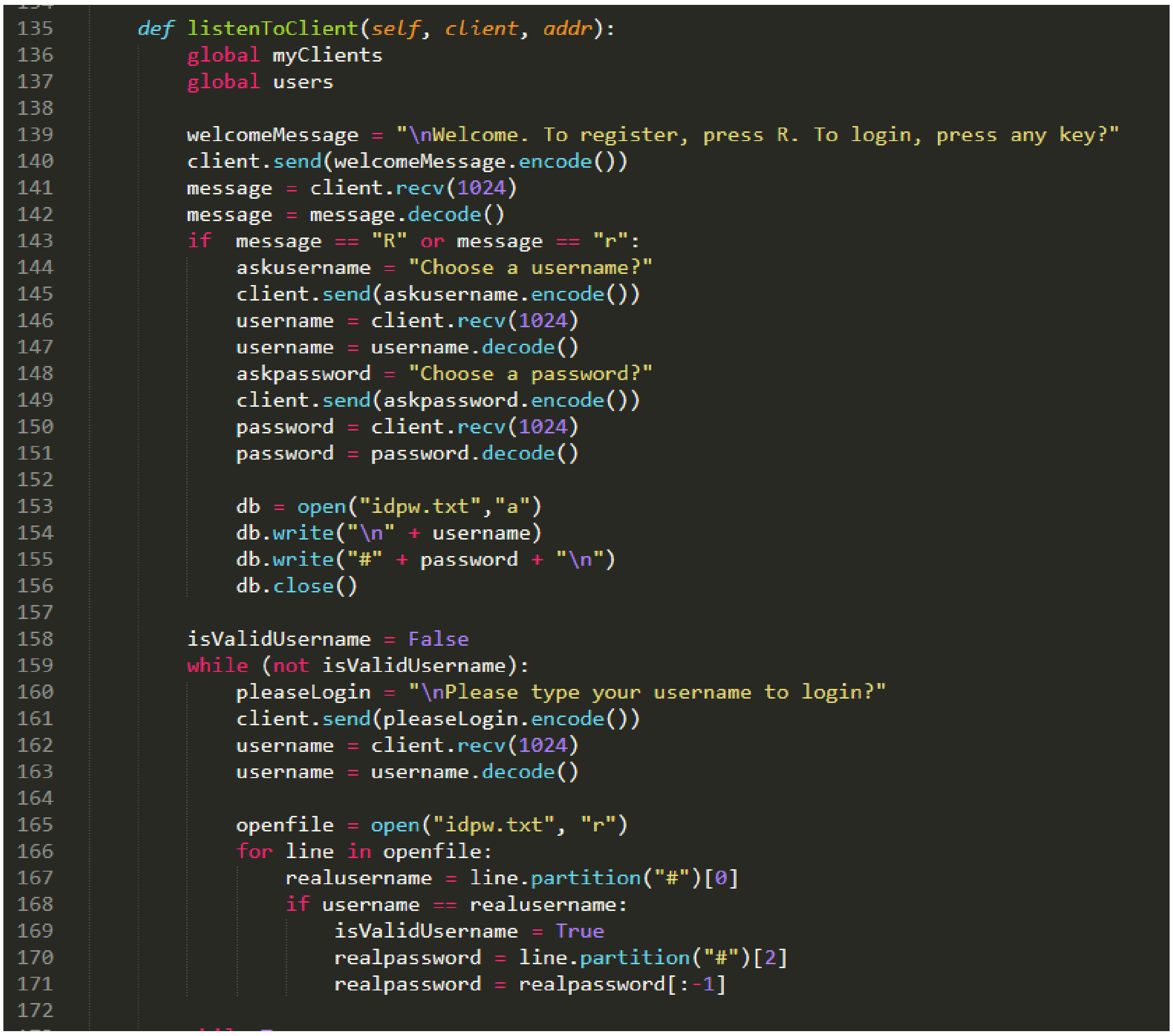
```
        while True:
            connectionSocket,addr=serverSocket.accept()
            myClients += [connectionSocket]
            threading.Thread(target=self.listenToClient, args = (connectionSocket,addr)).start()
```

Init function is quite basic so I only included end of the init function where there is a while loop. In this while loop, server accepts connections consistently. Clients that are connected are added to myClients list to keep track of them. And for every client, new thread is created and clients with the new threads are sent to listenToClient function.

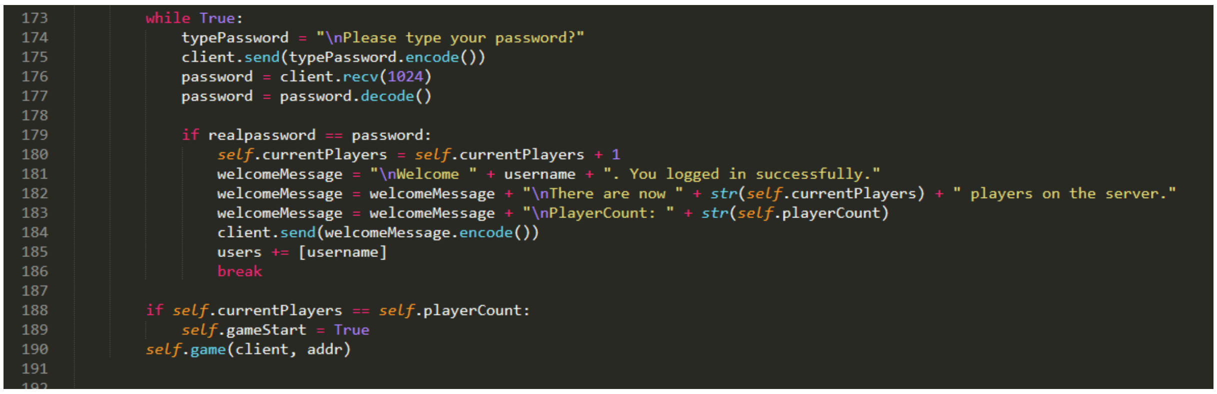
listenToClient Function (Line 135th)

In the first part (see the first image below) of the function, users are welcomed when they enter the server. They can register, or they can login if they registered before. Usernames and passwords are kept in a txt file called **“idpw.txt”** in plain text format. Clients can not enter the game unless they log in with a proper username. In the second part (second image below), password is asked and if they match, they enter with a proper username.

At the 188th line, we check if the number of users in the server is equal to the number of users needed for game to start. If they are equal, we set **gamestart = True** and if they are not, we just jump to the game function where all the fun begins.



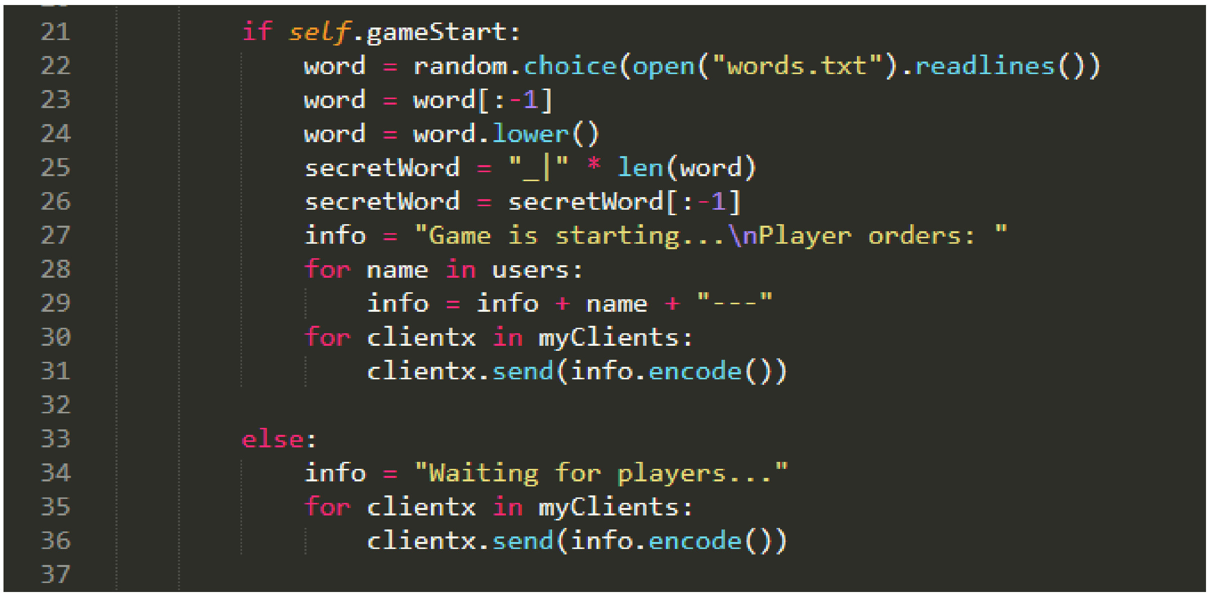
```
135     def listenToClient(self, client, addr):
136         global myClients
137         global users
138 
139         welcomeMessage = "\nWelcome. To register, press R. To login, press any key?"
140         client.send(welcomeMessage.encode())
141         message = client.recv(1024)
142         message = message.decode()
143         if  message == "R" or message == "r":
144             askusername = "Choose a username?"
145             client.send(askusername.encode())
146             username = client.recv(1024)
147             username = username.decode()
148             askpassword = "Choose a password?"
149             client.send(askpassword.encode())
150             password = client.recv(1024)
151             password = password.decode()
152 
153             db = open("idpw.txt","a")
154             db.write("\n" + username)
155             db.write("#" + password + "\n")
156             db.close()
157 
158         isValidUsername = False
159         while (not isValidUsername):
160             pleaseLogin = "\nPlease type your username to login?"
161             client.send(pleaseLogin.encode())
162             username = client.recv(1024)
163             username = username.decode()
164 
165             openfile = open("idpw.txt", "r")
166             for line in openfile:
167                 realusername = line.partition("#")[0]
168                 if username == realusername:
169                     isValidUsername = True
170                     realpassword = line.partition("#")[2]
171                     realpassword = realpassword[:-1]
172 
```



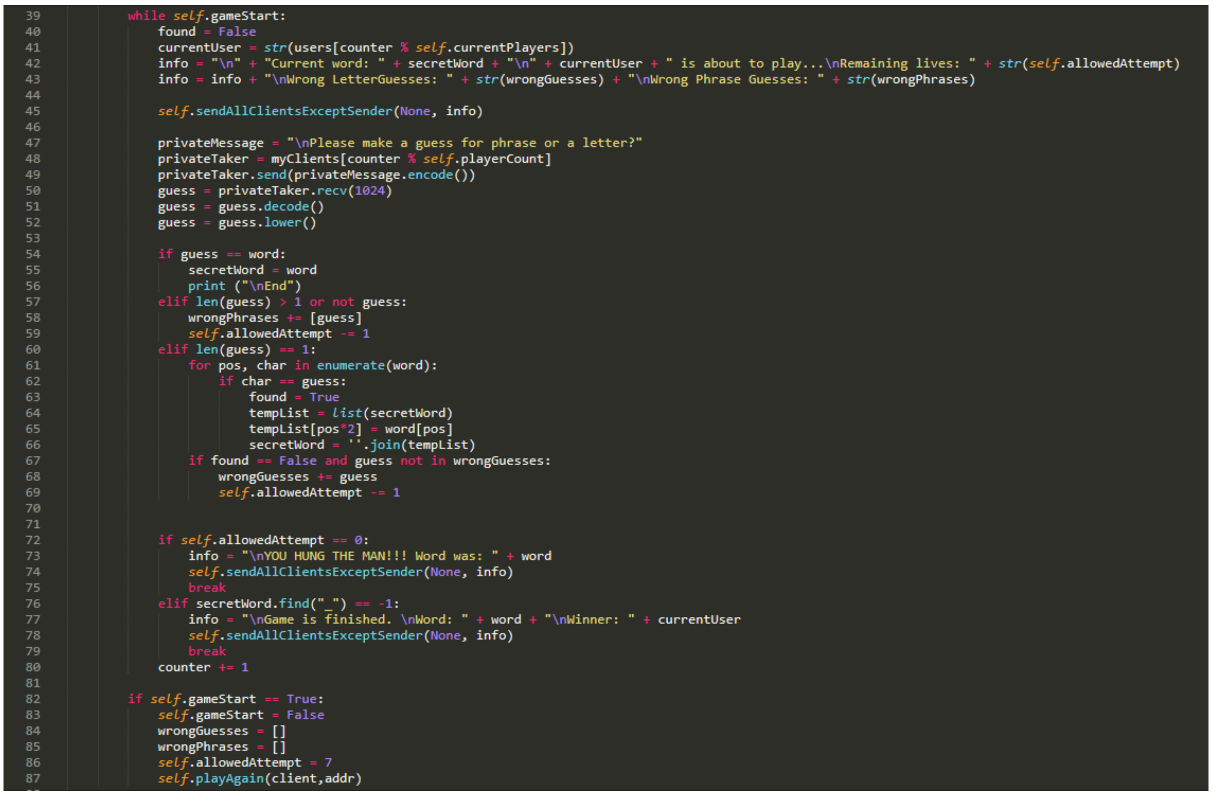
```
173         while True:
174             typePassword = "\nPlease type your password?"
175             client.send(typePassword.encode())
176             password = client.recv(1024)
177             password = password.decode()
178 
179             if realpassword == password:
180                 self.currentPlayers = self.currentPlayers + 1
181                 welcomeMessage = "\nWelcome " + username + ". You logged in successfully."
182                 welcomeMessage = welcomeMessage + "\nThere are now " + str(self.currentPlayers) + " players on the server."
183                 welcomeMessage = welcomeMessage + "\nPlayerCount: " + str(self.playerCount)
184                 client.send(welcomeMessage.encode())
185                 users += [username]
186                 break
187 
188         if self.currentPlayers == self.playerCount:
189             self.gameStart = True
190         self.game(client, addr)
191 
```

game function (Line 15th)

From the **21th line to 36th line, pre-game arrangement** is done such as selecting a random word and informing players about the order.



```
21          if self.gameStart:
22              word = random.choice(open("words.txt").readlines())
23              word = word[:-1]
24              word = word.lower()
25              secretWord = "_|" * len(word)
26              secretWord = secretWord[:-1]
27              info = "Game is starting...\nPlayer orders: "
28              for name in users:
29                  info = info + name + "---"
30              for clientx in myClients:
31                  clientx.send(info.encode())
32
33          else:
34              info = "Waiting for players..."
35              for clientx in myClients:
36                  clientx.send(info.encode())
37
```

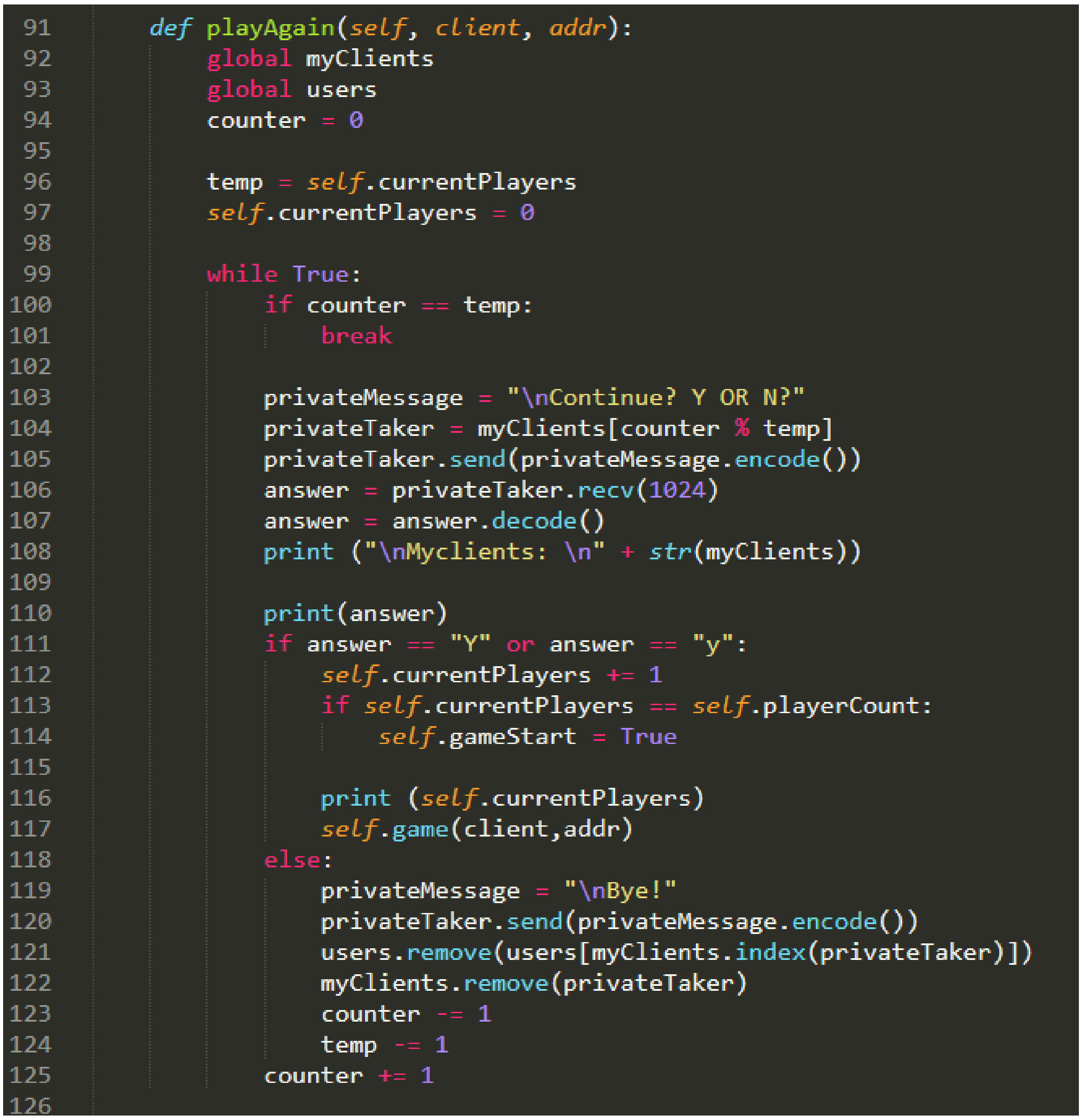


```
39          while self.gameStart:
40              found = False
41              currentUser = str(users[counter % self.currentPlayers])
42              info = "\n" + "Current word: " + secretWord + "\n" + currentUser + " is about to play...\nRemaining lives: " + str(self.allowedAttempt)
43              info = info + "\nWrong LetterGuesses: " + str(wrongGuesses) + "\nWrong Phrase Guesses: " + str(wrongPhrases)
44
45              self.sendAllClientsExceptSender(None, info)
46
47              privateMessage = "\nPlease make a guess for phrase or a letter?"
48              privateTaker = myClients[counter % self.playerCount]
49              privateTaker.send(privateMessage.encode())
50              guess = privateTaker.recv(1024)
51              guess = guess.decode()
52              guess = guess.lower()
53
54              if guess == word:
55                  secretWord = word
56                  print ("\nEnd")
57              elif len(guess) > 1 or not guess:
58                  wrongPhrases += [guess]
59                  self.allowedAttempt -= 1
60              elif len(guess) == 1:
61                  for pos, char in enumerate(word):
62                      if char == guess:
63                          found = True
64                          tempList = list(secretWord)
65                          tempList[pos*2] = word[pos]
66                          secretWord = ''.join(tempList)
67                  if found == False and guess not in wrongGuesses:
68                      wrongGuesses += guess
69                      self.allowedAttempt -= 1
70
71
72              if self.allowedAttempt == 0:
73                  info = "\nYOU HUNG THE MAN!!! Word was: " + word
74                  self.sendAllClientsExceptSender(None, info)
75                  break
76              elif secretWord.find("_") == -1:
77                  info = "\nGame is finished. \nWord: " + word + "\nWinner: " + currentUser
78                  self.sendAllClientsExceptSender(None, info)
79                  break
80              counter += 1
81
82          if self.gameStart == True:
83              self.gameStart = False
84              wrongGuesses = []
85              wrongPhrases = []
86              self.allowedAttempt = 7
87              self.playAgain(client,addr)
```

From the 39th line to 87th line, game is played. **Currentuser** holds the username of the user who is about to play and **privateTaker** holds the client information about the user who is about to play. Common messages are sent to clients by the **sendAllClientsExceptSender function**. First parameter of this function implies the client who the message will not be sent. If it is None, then message is sent to everyone. Game states are changed among these lines as well as game termination.

playAgain function (Line 91th)

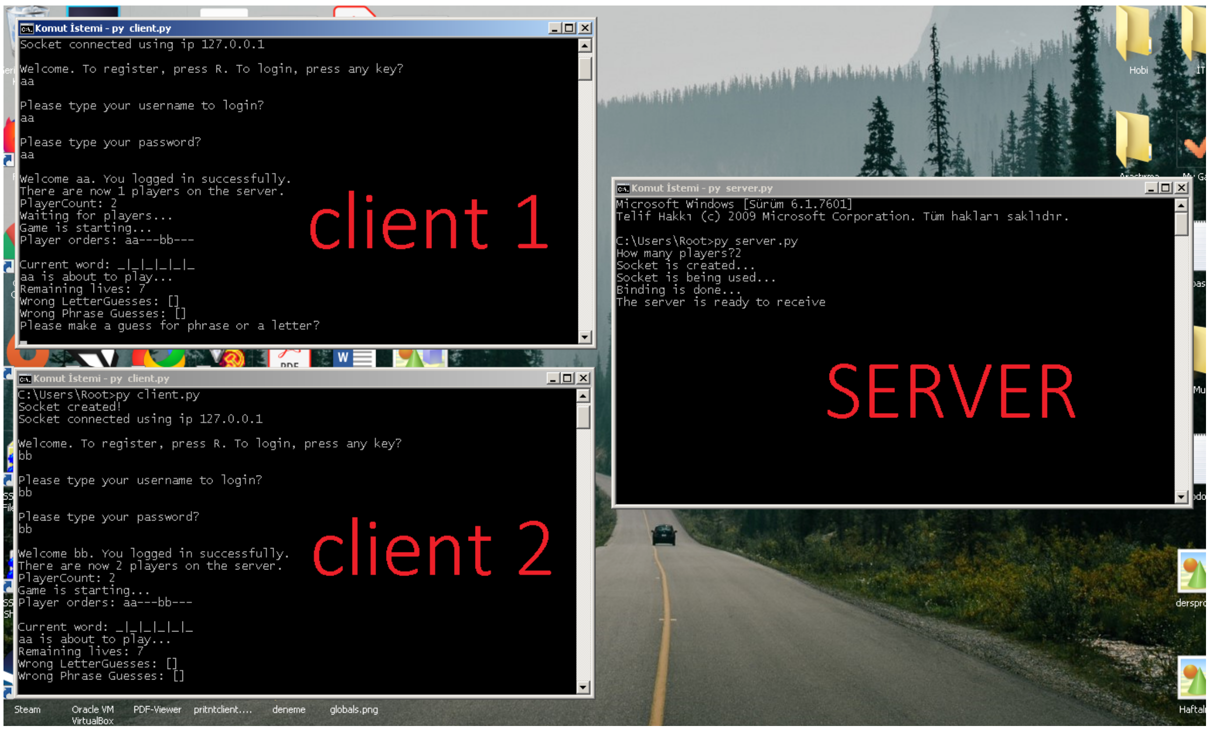
This function asks user if they want to continue playing or not. If they don't want to play, they are removed from both users and clients list. If they want to keep continue, they are moved to the game function where they will have to wait for needed number of clients to connect.



```
91      def playAgain(self, client, addr):
92          global myClients
93          global users
94          counter = 0
95  
96          temp = self.currentPlayers
97          self.currentPlayers = 0
98  
99          while True:
100             if counter == temp:
101                 break
102 
103             privateMessage = "\nContinue? Y OR N?"
104             privateTaker = myClients[counter % temp]
105             privateTaker.send(privateMessage.encode())
106             answer = privateTaker.recv(1024)
107             answer = answer.decode()
108             print ("\nMyclients: \n" + str(myClients))
109 
110             print(answer)
111             if answer == "Y" or answer == "y":
112                 self.currentPlayers += 1
113                 if self.currentPlayers == self.playerCount:
114                     self.gameStart = True
115 
116                 print (self.currentPlayers)
117                 self.game(client,addr)
118             else:
119                 privateMessage = "\nBye!"
120                 privateTaker.send(privateMessage.encode())
121                 users.remove(users[myClients.index(privateTaker)])
122                 myClients.remove(privateTaker)
123                 counter -= 1
124                 temp -= 1
125             counter += 1
126 
```

2) Screenshots from the game

a) When game just started

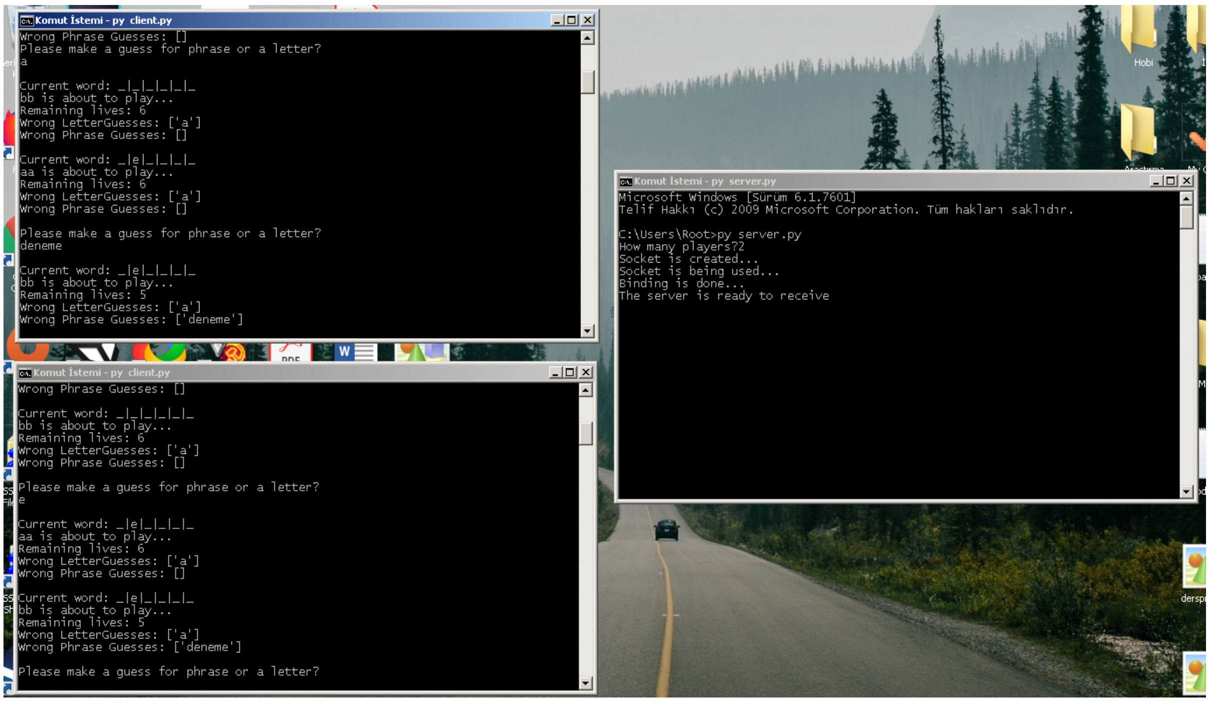

Komut İstemi - py client.py
client 1
Komut İstemi - py client.py
client 2
Komut İstemi - py server.py
SERVER

```
Socket connected using ip 127.0.0.1

Welcome. To register, press R. To login, press any key?
aa

Please type your username to login?
aa

Please type your password?
aa

Welcome aa. You logged in successfully.
There are now 1 players on the server.
PlayerCount: 2
Waiting for players...
Game is starting...
Player orders: aa---bb---

Current word: _|_|_|_|_|_
aa is about to play...
Remaining lives: 7
Wrong LetterGuesses: []
Wrong Phrase Guesses: []
Please make a guess for phrase or a letter?
```

```
C:\Users\Root>py client.py
Socket created!
Socket connected using ip 127.0.0.1

Welcome. To register, press R. To login, press any key?
bb

Please type your username to login?
bb

Please type your password?
bb

Welcome bb. You logged in successfully.
There are now 2 players on the server.
PlayerCount: 2
Game is starting...
Player orders: aa---bb---

Current word: _|_|_|_|_|_
aa is about to play...
Remaining lives: 7
Wrong LetterGuesses: []
Wrong Phrase Guesses: []
```

```
Microsoft Windows [Sürüm 6.1.7601]
Telif Hakkı (c) 2009 Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Root>py server.py
How many players?2
Socket is created...
Socket is being used...
Binding is done...
The server is ready to receive
```

b) Middle game

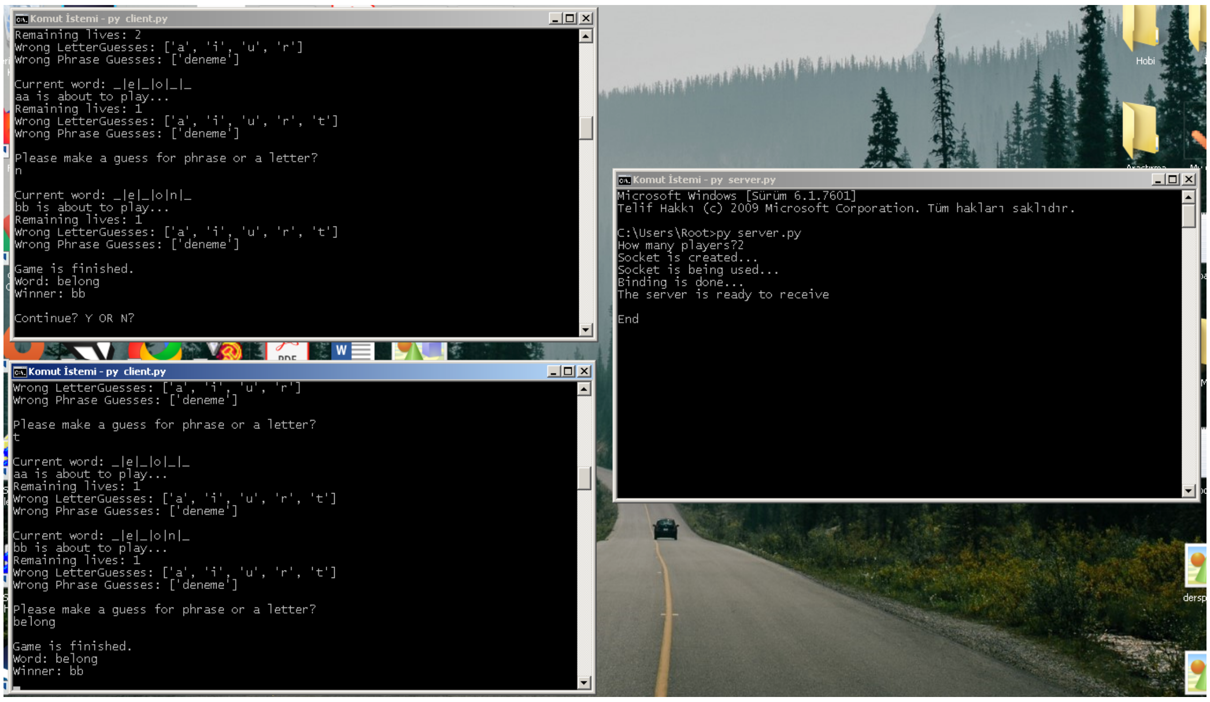

Komut İstemi - py client.py
Komut İstemi - py client.py
Komut İstemi - py server.py

```
Wrong Phrase Guesses: []
Please make a guess for phrase or a letter?
a

Current word: _|_|_|_|_|_
bb is about to play...
Remaining lives: 6
Wrong LetterGuesses: ['a']
Wrong Phrase Guesses: []

Current word: _|e|_|_|_|_
aa is about to play...
Remaining lives: 6
Wrong LetterGuesses: ['a']
Wrong Phrase Guesses: []

Please make a guess for phrase or a letter?
deneme

Current word: _|e|_|_|_|_
bb is about to play...
Remaining lives: 5
Wrong LetterGuesses: ['a']
Wrong Phrase Guesses: ['deneme']
```

```
Wrong Phrase Guesses: []

Current word: _|_|_|_|_|_
bb is about to play...
Remaining lives: 6
Wrong LetterGuesses: ['a']
Wrong Phrase Guesses: []

Please make a guess for phrase or a letter?
e

Current word: _|e|_|_|_|_
aa is about to play...
Remaining lives: 6
Wrong LetterGuesses: ['a']
Wrong Phrase Guesses: []

Current word: _|e|_|_|_|_
bb is about to play...
Remaining lives: 5
Wrong LetterGuesses: ['a']
Wrong Phrase Guesses: ['deneme']

Please make a guess for phrase or a letter?
```

```
Microsoft Windows [Sürüm 6.1.7601]
Telif Hakkı (c) 2009 Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Root>py server.py
How many players?2
Socket is created...
Socket is being used...
Binding is done...
The server is ready to receive
```

c) Game finish with success



Komut İstemi - py client.py

```
Remaining lives: 2
Wrong LetterGuesses: ['a', 'i', 'u', 'r']
Wrong Phrase Guesses: ['deneme']

Current word: _|e|_|o|_|_
aa is about to play...
Remaining lives: 1
Wrong LetterGuesses: ['a', 'i', 'u', 'r', 't']
Wrong Phrase Guesses: ['deneme']

Please make a guess for phrase or a letter?
n

Current word: _|e|_|o|n|_
bb is about to play...
Remaining lives: 1
Wrong LetterGuesses: ['a', 'i', 'u', 'r', 't']
Wrong Phrase Guesses: ['deneme']

Game is finished.
Word: belong
Winner: bb

Continue? Y OR N?
```

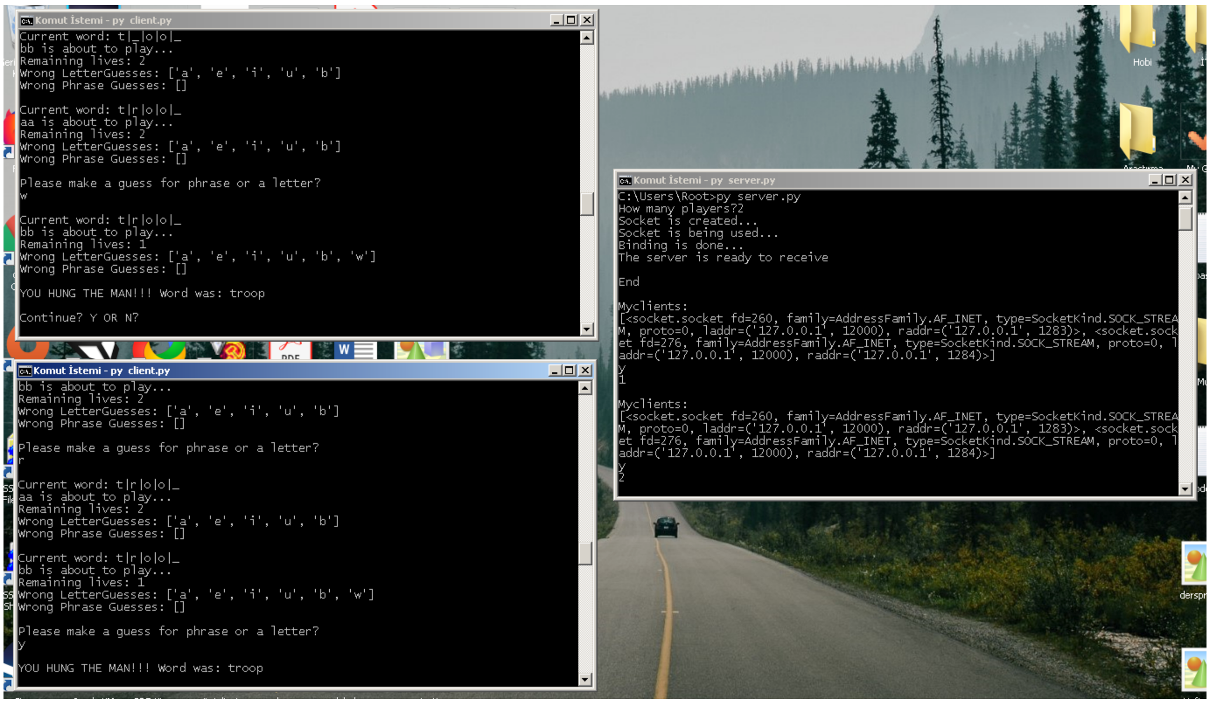
Komut İstemi - py client.py

```
Wrong LetterGuesses: ['a', 'i', 'u', 'r']
Wrong Phrase Guesses: ['deneme']

Please make a guess for phrase or a letter?
t

Current word: _|e|_|o|_|_
aa is about to play...
Remaining lives: 1
Wrong LetterGuesses: ['a', 'i', 'u', 'r', 't']
Wrong Phrase Guesses: ['deneme']

Current word: _|e|_|o|n|_
bb is about to play...
Remaining lives: 1
Wrong LetterGuesses: ['a', 'i', 'u', 'r', 't']
Wrong Phrase Guesses: ['deneme']

Please make a guess for phrase or a letter?
belong

Game is finished.
Word: belong
Winner: bb
```

Komut İstemi - py server.py

```
Microsoft Windows [Sürüm 6.1.7601]
Telif Hakkı (c) 2009 Microsoft Corporation. Tüm hakları saklıdır.

C:\Users\Root>py server.py
How many players?2
Socket is created...
Socket is being used...
Binding is done...
The server is ready to receive

End
```

d) Game finish with failure



Komut İstemi - py client.py

```
Current word: t|_|o|o|_
bb is about to play...
Remaining lives: 2
Wrong LetterGuesses: ['a', 'e', 'i', 'u', 'b']
Wrong Phrase Guesses: []

Current word: t|r|o|o|_
aa is about to play...
Remaining lives: 2
Wrong LetterGuesses: ['a', 'e', 'i', 'u', 'b']
Wrong Phrase Guesses: []

Please make a guess for phrase or a letter?
w

Current word: t|r|o|o|_
bb is about to play...
Remaining lives: 1
Wrong LetterGuesses: ['a', 'e', 'i', 'u', 'b', 'w']
Wrong Phrase Guesses: []

YOU HUNG THE MAN!!! Word was: troop

Continue? Y OR N?
```

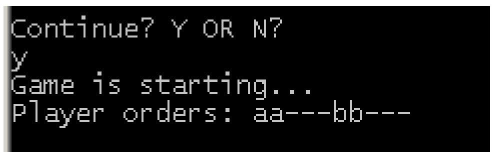
Komut İstemi - py client.py

```
bb is about to play...
Remaining lives: 2
Wrong LetterGuesses: ['a', 'e', 'i', 'u', 'b']
Wrong Phrase Guesses: []

Please make a guess for phrase or a letter?
r

Current word: t|r|o|o|_
aa is about to play...
Remaining lives: 2
Wrong LetterGuesses: ['a', 'e', 'i', 'u', 'b']
Wrong Phrase Guesses: []

Current word: t|r|o|o|_
bb is about to play...
Remaining lives: 1
Wrong LetterGuesses: ['a', 'e', 'i', 'u', 'b', 'w']
Wrong Phrase Guesses: []

Please make a guess for phrase or a letter?
y

YOU HUNG THE MAN!!! Word was: troop
```

Komut İstemi - py server.py

```
C:\Users\Root>py server.py
How many players?2
Socket is created...
Socket is being used...
Binding is done...
The server is ready to receive

End

Myclients:
[<socket.socket fd=260, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREA
M, proto=0, laddr=('127.0.0.1', 12000), raddr=('127.0.0.1', 1283)>, <socket.sock
et fd=276, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, l
addr=('127.0.0.1', 12000), raddr=('127.0.0.1', 1284)>]
y
1

Myclients:
[<socket.socket fd=260, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREA
M, proto=0, laddr=('127.0.0.1', 12000), raddr=('127.0.0.1', 1283)>, <socket.sock
et fd=276, family=AddressFamily.AF_INET, type=SocketKind.SOCK_STREAM, proto=0, l
addr=('127.0.0.1', 12000), raddr=('127.0.0.1', 1284)>]
y
2
```

e) Asking if they want to continue



```
Continue? Y OR N?
y
Game is starting...
Player orders: aa---bb---
```