PeerToPeer-UDP

Input :

udp_s.py

```
#!/usr/bin/env python

"""
A simple echo server
"""
import socket
from threading import Thread, Lock
import sys
def receiver(sock):
	global flag
	while flag:
		data=sock.recv(size) #receive client data
		'''Receive data from client''' #Receive client data
		if data == 'quit':
			sys.exit(0)
		print "\t\t"+data
	flag = True
	print "Thread Exiting"

host = '172.19.229.227'		#Server IP addres
port = 50004	#Port address
backlog = 5	#For binding connections
size = 1024	#Size of packet data to receive
flag = True

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM) #Creating a Socket
s.bind((host,port))	#Binding Port and IP address
s.listen(backlog)		#Listen to connection

while 1:
	print "Waiting for Peer to Join..."
	client,address=s.accept()
	'''Accept a connection'''	#Accept a client connecion
	print "Client "+str(address)+"joined chat.\n"

	# Start channel receiver thread
	'''Create a thread'''
	recv_thread=Thread(target=receiver,args=(client,))
	recv_thread.daemon = True
```

```
	recv_thread.start()

	data = "random"		#Entering any value so it is not quit initially
	while data != 'quit':	#User will enter 'quit' to exit server
		data = raw_input()
		client.send(data)	#Enter message to send to client
		'''Send data to client'''	#Sending data to client
	flag = False
	client.close()	#Closing the socket
```

udp_c.py

```
#!/usr/bin/env python

"""
A simple peer to peer chat
lines: 14
"""

import socket
from threading import Thread, Lock
import sys

def receiver(sock):
	while flag:
		data = sock.recv(1024) #Receive data from server
		if data == 'quit':
			sys.exit(0)
		print "\t\t"+data

host = '172.19.229.227'		#Server IP addres
port = 50004			#Port address
size = 1024			#Size of packet data to receive
flag = True

sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)	#Creating a Socket
sock.connect((host,port))	#Connect to Server

# Start Receiver Thread
hop = Thread(target=receiver, args=(sock,))
hop.daemon = True
hop.start()

data = "random"		#Entering any value so it is not quit initially

while data != 'quit':	#User will enter 'quit' to exit server
```

```
        data = raw_input()#Enter message to send to server
        sock.send(data)
        '''Send data to server''' #Sending data to server
sock.close()    #Closing the socket
```

Output :