190905104
Parth Shukla

Write a UDP time server to display the current time and day.

Server

```
import socket
import time

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM) # For UDP
udp_host = socket.gethostname() # Host IP
udp_port = 12345 # specified port to connect
sock.bind((udp_host, udp_port))
currentTime = time.ctime(time.time()) + "\r\n"

while True:
    print ("Waiting for client...")
    data,addr = sock.recvfrom(1024) #receive data from client
    print ("Received Messages:",data.decode()," from",addr)
    currentTime = time.ctime(time.time()) + "\r\n"
    sock.sendto(currentTime.encode('ascii'), addr)
```

Client

```
import socket

s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

HOST = socket.gethostname()
PORT = 12345

msg = 'connected'
s.sendto(msg.encode(),(HOST,PORT))

t = s.recv(1024)
print('Time: ', t.decode())
s.close()
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 udptimeser
v.py
Waiting for client...
Received Messages: connected  from ('127.0.0.1', 58282)
Waiting for client...
```

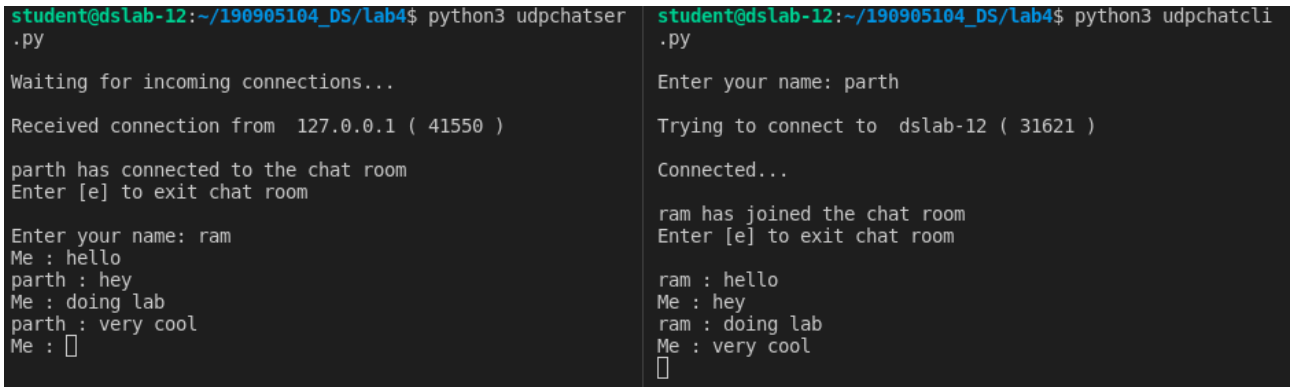
```
student@dslab-12:~/190905104_DS/lab4$ python3 udptimecli
.py
Time:  Thu Mar 31 10:19:23 2022

student@dslab-12:~/190905104_DS/lab4$
```

Write a UDP simple chat program for message send and receive

Server

```
import socket

HOST = socket.gethostname()
PORT = 31621

s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

s.bind((HOST, PORT))
print("\nWaiting for incoming connections...\n")
data,addr = s.recvfrom(1024)
```

```
print("Received connection from ", addr[0], "(", addr[1], ")\n")
s_name, addr = s.recvfrom(1024)
s_name = s_name.decode()
print(s_name, "has connected to the chat room\nEnter [e] to exit chat room\n")
name = input(str("Enter your name: "))
s.sendto(name.encode('ascii') , addr)
while True:
    message = input(str("Me : "))
    if message == "[e]":
        message = "Left chat room!"
        s.sendto(message.encode() , addr)
        print("\n")
        break
    s.sendto(message.encode() , addr)
    message, addr = s.recvfrom(1024)
    message = message.decode()
    print(s_name, ":", message)

# Client

import socket
HOST = socket.gethostname()
PORT = 31621 # Port to listen on (non-privileged ports are > 1023)
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
name = input(str("\nEnter your name: "))
print("\nTrying to connect to ", HOST, "(", PORT, ")\n")
s.sendto(b"" , (HOST , PORT))
print("Connected...\n")
s.sendto(name.encode('ascii') , (HOST , PORT))
s_name = s.recv(1024)
s_name = s_name.decode()
print(s_name, "has joined the chat room\nEnter [e] to exit chat room\n")
while True:
    message, addr = s.recvfrom(1024)
    message = message.decode()
    print(s_name, ":", message)
    message = input(str("Me : "))
    if message == "[e]":
        message = "Left chat room!"
        s.sendto(message.encode() , (HOST , PORT))
        print("\n")
        break
    s.sendto(message.encode() , (HOST , PORT))
```



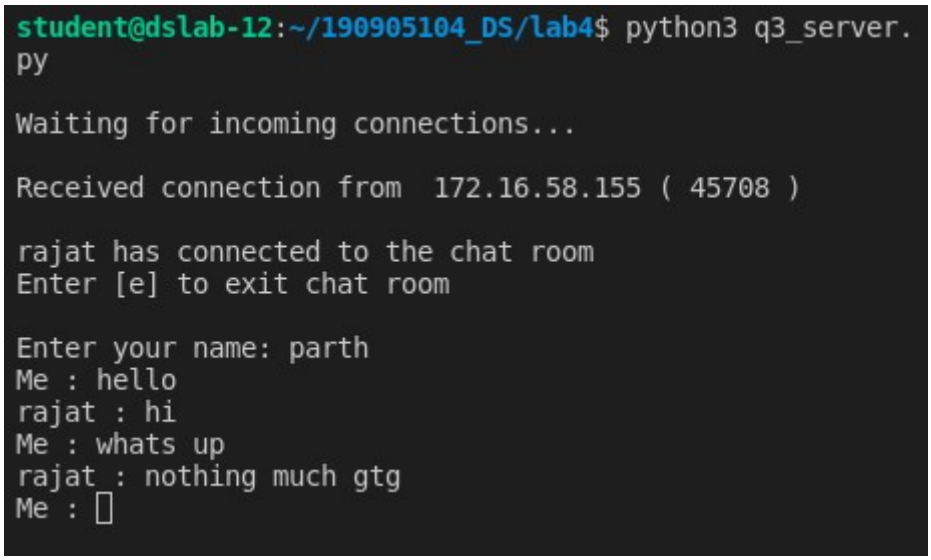
Write a TCP/UDP peer to peer chat system between two different machines

Server

```
import socket
HOST = '172.16.58.91' # Standard loopback interface address (localhost)
PORT = 31621 # Port to listen on (non-privileged ports are > 1023)
s = socket.socket()
s.bind((HOST, PORT))
s.listen()
print("\nWaiting for incoming connections...\n")
conn, addr = s.accept()
print("Received connection from ", addr[0], "(", addr[1], ")\n")
s_name = conn.recv(1024)
s_name = s_name.decode()
print(s_name, "has connected to the chat room\nEnter [e] to exit chat room\n")
name = input(str("Enter your name: "))
conn.send(name.encode())
while True:
    message = input(str("Me : "))
    if message == "[e]":
        message = "Left chat room!"
        conn.send(message.encode())
        print("\n")
        break
    conn.send(message.encode())
    message = conn.recv(1024)
    message = message.decode()
    print(s_name, ":", message)
```

Client

```
import socket
HOST = '172.16.58.155' # Standard loopback interface address (localhost)
PORT = 31621 # Port to listen on (non-privileged ports are > 1023)
s = socket.socket()
name = input(str("\nEnter your name: "))
print("\nTrying to connect to ", HOST, "(", PORT, ")\n")
s.connect((HOST, PORT))
print("Connected...\n")
s.send(name.encode())
s_name = s.recv(1024)
s_name = s_name.decode()
print(s_name, "has joined the chat room\nEnter [e] to exit chat room\n")
while True:
    message = s.recv(1024)
    message = message.decode()
    print(s_name, ":", message)
    message = input(str("Me : "))
    if message == "[e]":
        message = "Left chat room!"
        s.send(message.encode())
        print("\n")
        break
    s.send(message.encode())
```

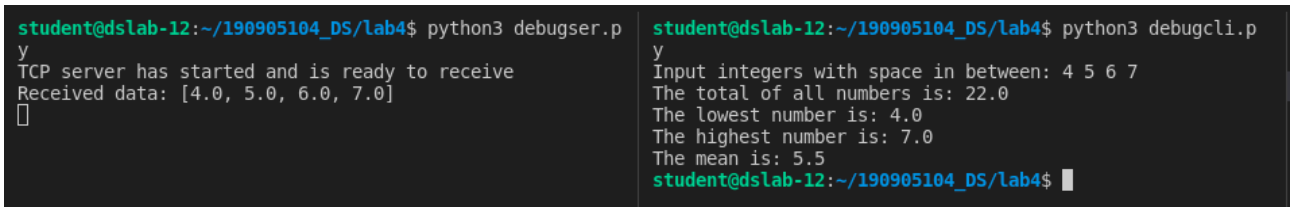


Debug question
Changes made -
1) Changed raw_input to input because we are using python3
2) Used brachets in print
3) Added decode() and encode() whereever there is communication
4) Removed the sending part for the length of the array as it was redundant and we are using string function split() to reconstruct the list.

```
# Server
import socket

serverIP = 'localhost'
serverPort = 16001

serverSock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
serverSock.bind((serverIP, serverPort))
serverSock.listen(1)

print("TCP server has started and is ready to receive")

while 1:
    connection, addr = serverSock.accept()
    data = connection.recv(1024)

    if not data: break

    temp = [float(x) for x in data.decode().split(' ')]
    print( "Received data:", temp )

    length = len(temp)
    maximum = max(temp)
    minimum = min(temp)
    total = sum(temp)
    mean = total/length
```

```
    msg = str(total) + " " + str(minimum) + " " + str(maximum) + " " + str(mean)


    connection.send(str(msg).encode())

# Client

import socket


serverIP = 'localhost'

serverPort = 16001


clientSock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

clientSock.connect((serverIP, serverPort))


message = input("Input integers with space in between: ")
# print(message)
# message2 = input("Enter the length of the set: ")


clientSock.send(message.encode())

# clientSock.send(message2.encode())


data = clientSock.recv(1024)


temp = [float(x) for x in data.decode().split(' ')]


print("The total of all numbers is: " + str(temp[0]))

print("The lowest number is: " + str(temp[1]))
```

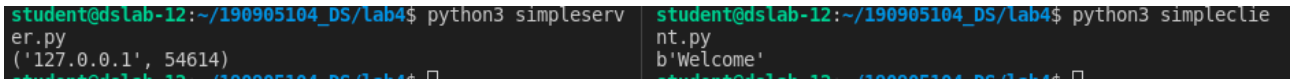
```
print("The highest number is: " + str(temp[2]))

print("The mean is: " + str(temp[3]))


clientSock.close()
```



Solved Examples

Simple client server

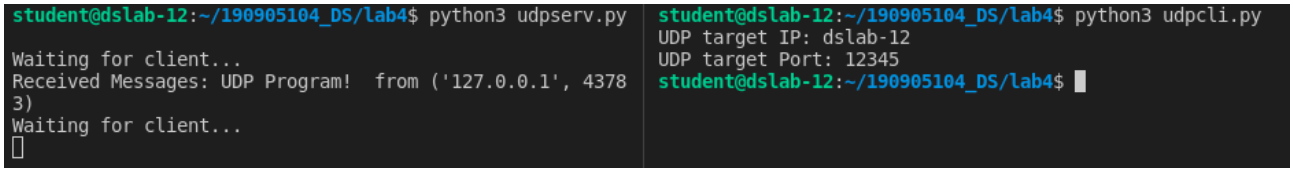
```
# Server
import socket
host = socket.gethostname()
port = 12345
s = socket.socket()
s.bind((host, port))
s.listen(5)
conn, addr = s.accept()
print(addr)
while True:
    data = conn.recv(1024)
    if not data:
        break
    conn.sendall(data)
conn.close()

# Client
import socket
host = socket.gethostname()
port = 12345
s = socket.socket()
s.connect((host, port))
s.sendall(b'Welcome')
data = s.recv(1024)
s.close()
print(repr(data))
```



Simple UDP

```
#Server
import socket

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM) # For UDP
udp_host = socket.gethostname() # Host IP
udp_port = 12345 # specified port to connect
sock.bind((udp_host, udp_port))
while True:
    print ("Waiting for client...")
    data,addr = sock.recvfrom(1024) #receive data from client
    print ("Received Messages:",data.decode()," from",addr)
```

```
# Client
import socket
sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM) # For UDP
udp_host = socket.gethostname() # Host IP
udp_port = 12345 # specified port to connect
msg = "UDP Program!"
print ("UDP target IP:", udp_host)
print ("UDP target Port:", udp_port)
sock.sendto(msg.encode(),(udp_host,udp_port))
```



Write a program where client can send a message to the server and the server can receive the message and send, or echo, it back to the client.

```
# Server
import socket
HOST = '127.0.0.1' # Standard loopback interface address (localhost)
PORT = 2053 # Port to listen on (non-privileged ports are > 1023)
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen()
    conn, addr = s.accept()
    with conn:
        print('Connected by', addr)
        while True:
            data = conn.recv(1024)
            if data:
                print("Client: ",data.decode())
                data = input("Enter message to client:")
```

```
            if not data:
                break
            # sending message as bytes to client.
            conn.sendall(bytearray(data, 'utf-8'))
        conn.close()

# Client
import socket
HOST = '127.0.0.1' # The server's hostname or IP address
PORT = 2053 # The port used by the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((HOST, PORT))
    s.sendall(b'Hello, world')
    data = s.recv(1024)
    print('Received Connection')
    print('Server:', data.decode())
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 echoserv.p
y
Connected by ('127.0.0.1', 33410)
Client:  Hello, world
Enter message to client:hello client
student@dslab-12:~/190905104_DS/lab4$
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 echocli.py

Received Connection
Server: hello client
student@dslab-12:~/190905104_DS/lab4$
```

Write a program to create TCP time server in Python

```
# Server

import socket
import time

HOST = '127.0.0.1' # Standard loopback interface address (localhost)
PORT = 2055

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen(5)
    while True:
        clientsock, addr = s.accept()
        print('Connecttion established')
        currentTime = time.ctime(time.time()) + "\r\n"
        clientsock.send(currentTime.encode('ascii'))
        clientsock.close()


# Client

import socket
HOST = '127.0.0.1' # The server's hostname or IP address
PORT = 2055 # The port used by the server

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((HOST, PORT))
    data = s.recv(1024)
    print('Received Connection')
    print('Time sent by server: ', data.decode())
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 timeserver
.py
Connecttion established
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 timeclient
.py
Received Connection
Time sent by server:  Thu Mar 31 09:33:55 2022
```

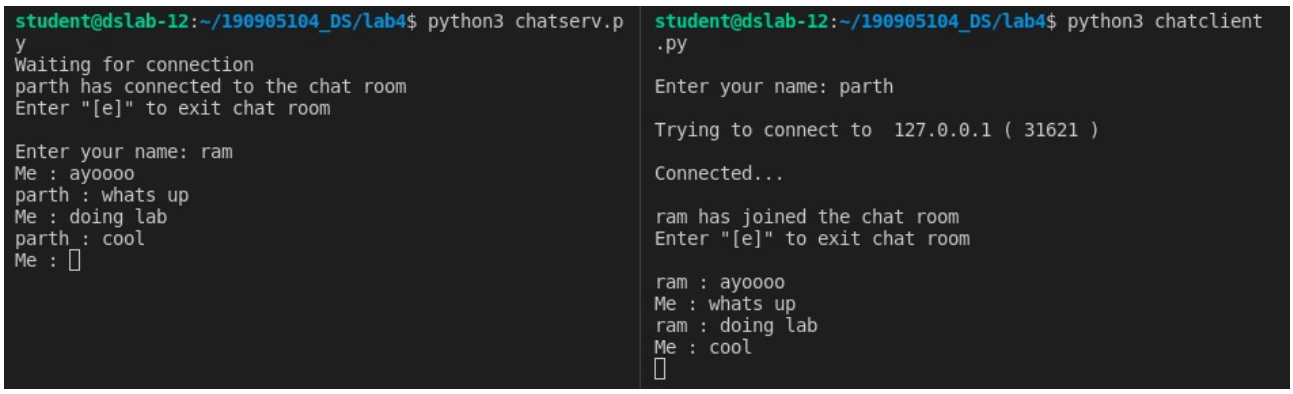
Write a TCP chat server in python using socket programming

Server

```
import socket
HOST = '127.0.0.1' # Standard loopback interface address (localhost)
PORT = 31621

s = socket.socket()
s.bind((HOST, PORT))
s.listen(5)
print('Waiting for connection')
conn, addr = s.accept()

s_name = conn.recv(1024)
s_name = s_name.decode()
print(s_name, "has connected to the chat room\nEnter \"[e]\" to exit chat room\n")
name = input(str("Enter your name: "))
conn.send(name.encode())
while True:
    message = input(str("Me : "))
    if message == "[e]":
        message = "Left chat room!"
        conn.send(message.encode())
        print("\n")
        break
    conn.send(message.encode())
    message = conn.recv(1024)
    message = message.decode()
    print(s_name, ":", message)
```

Client

```
import socket
HOST = '127.0.0.1' # Standard loopback interface address (localhost)
PORT = 31621 # Port to listen on (non-privileged ports are > 1023)
s = socket.socket()
name = input(str("\nEnter your name: "))
print("\nTrying to connect to ", HOST, "(", PORT, ")\n")
s.connect((HOST, PORT))
print("Connected...\n")
s.send(name.encode())
s_name = s.recv(1024)
s_name = s_name.decode()
print(s_name, "has joined the chat room\nEnter \"[e]\" to exit chat room\n")
while True:
    message = s.recv(1024)
    message = message.decode()
    print(s_name, ":", message)
    message = input(str("Me : "))
    if message == "[e]":
        message = "Left chat room!"
        s.send(message.encode())
        print("\n")
        break
    s.send(message.encode())
```



Threading

```
# Server

import socket
import os
from _thread import *

HOST = '127.0.0.1'
PORT = 5555

s = socket.socket()

print('Waiting for connection')

tcount = 0
try:
    s.bind((HOST, PORT))
except error as e:
    print(e)

s.listen(5)

def tclient(conn):
    conn.send(str.encode('Welcome to server'))
    while True:
        data = conn.recv(1024)
        print('Received from client :' + str(tcount) +data.decode())
        inp = input('Server says: ')
        if not data:
            break
        conn.sendall(inp.encode())
    conn.close()

while True:
    cli, addr = s.accept()
    print('Connected to: ' + addr[0] + ':' + str(addr[1]))
    start_new_thread(tclient, (cli, ))
    tcount+=1
    print('Thread count= ', tcount)
s.close()

# Client

import socket
ClientSocket = socket.socket()
host = '127.0.0.1'
port = 5555
print('Waiting for connection')
```

```
try:
    ClientSocket.connect((host, port))
except socket.error as e:
    print(str(e))
Response = ClientSocket.recv(1024)
while True:
    Input = input('Client Say Something: ')
    ClientSocket.send(str.encode(Input))
    Response = ClientSocket.recv(1024)
    print('From Server : ' + Response.decode())
ClientSocket.close()
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 conserv.py

Waiting for connection
Connected to: 127.0.0.1:57132
Thread count=  1
Received from client :1first client
Server says: copy
Connected to: 127.0.0.1:57134
Thread count=  2
Received from client :2second client
Server says:
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 concli.py
Waiting for connection
[Errno 111] Connection refused
Traceback (most recent call last):
  File "concli.py", line 10, in <module>
    Response = ClientSocket.recv(1024)
OSError: [Errno 107] Transport endpoint is not connected
student@dslab-12:~/190905104_DS/lab4$ python3 concli.py
Waiting for connection
Client Say Something: first client
From Server : copy
Client Say Something:
```

```
student@dslab-12:~/190905104_DS/lab4$ python3 concli.py
Waiting for connection
Client Say Something: second client
```