LAB – 4
SOCKET PROGRAMMING USING PYTHON

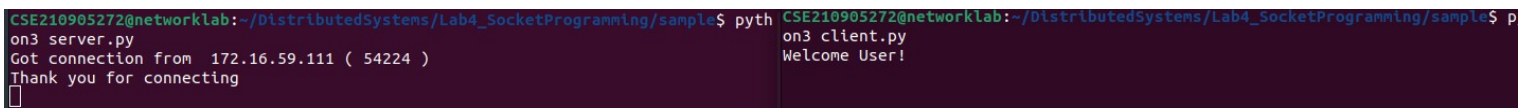
CONNECTION ORIENTED:

server.py

```
import socket
host = socket.gethostname()
port=12345
s=socket.socket()
s.bind((host,port))
s.listen(5)
conn,addr = s.accept()
print('Got connection from ',addr[0], '(', addr[1], ')')
print('Thank you for connecting')
while True:
        data=conn.recv(1024)
        if not data: break
        conn.sendall(data)
conn.close()
```

client.py

```
import socket
host = socket.gethostname()
port=12345
s=socket.socket()
s.connect((host,port))

print('Welcome User!')
data=s.recv(1024)
s.close()
```



CONNECTION LESS:

server.py

```
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.py

```
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))
```

.

Q)# 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

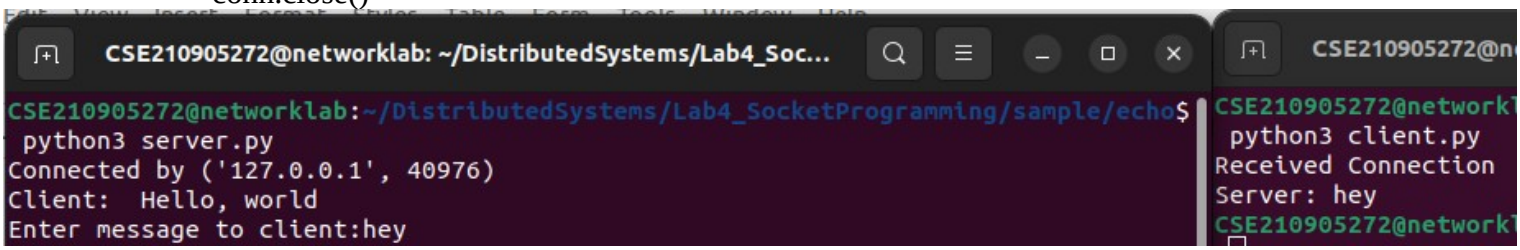
client.py

```
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())
```

server.py

```
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()
```



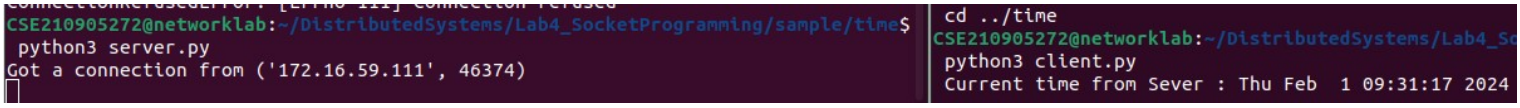
Q)**Write a program to create TCP time server in Python**

server.py

```
import socket
import time

# create a socket object
serversocket = socket.socket(
socket.AF_INET, socket.SOCK_STREAM)
# get local machine name
host = socket.gethostname()
port = 9991
# bind to the port
serversocket.bind((host, port))
# queue up to 5 requests
serversocket.listen(5)
while True:
# establish a connection
        clientsocket,addr = serversocket.accept()
        print("Got a connection from %s" % str(addr))
        currentTime = time.ctime(time.time()) + "\r\n"
        clientsocket.send(currentTime.encode('ascii'))
        clientsocket.close()
```

client.py

```
import socket
# create a socket object
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
# get local machine name
host = socket.gethostname()
port = 9991
# connection to hostname on the port.
s.connect((host, port))
# Receive no more than 1024 bytes
tm = s.recv(1024)
print(' Current time from Sever :', tm.decode())
s.close()
```



Q)**Write a TCP chat server in python using socket programming.**

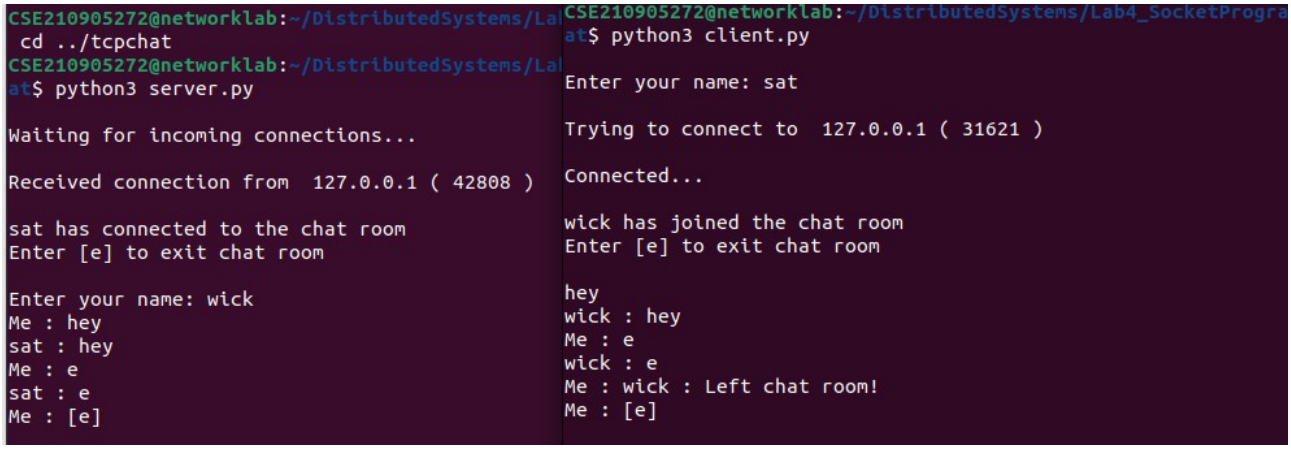
Server.py

```
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()
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.py

```
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())
```



Q)Forking/ Threading (Concurrent Server)

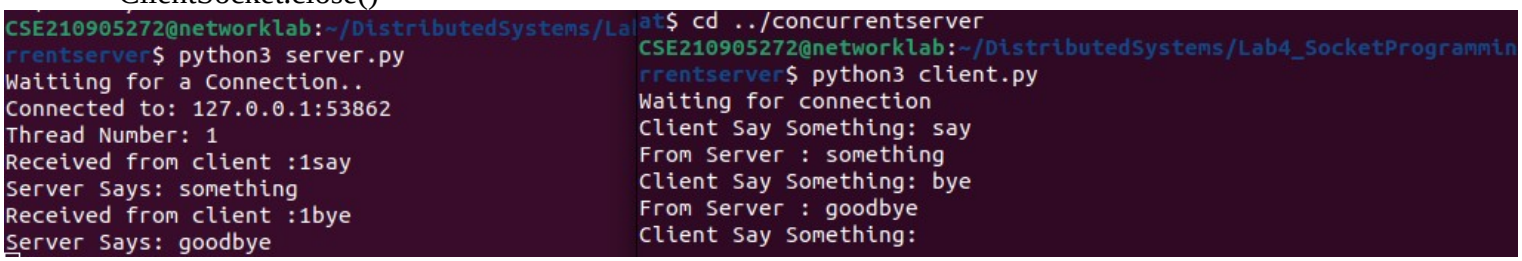
server.py

```
import socket
import os
from _thread import *
ServerSocket = socket.socket()
host = '127.0.0.1'
port = 11596
ThreadCount = 0
try:
        ServerSocket.bind((host, port))
except socket.error as e:
        print(str(e))
print('Waitiing for a Connection..')
ServerSocket.listen(5)
def threaded_client(connection):
        connection.send(str.encode('Welcome to the Server'))
        while True:
                data = connection.recv(2048)
                print('Received from client :' + str(ThreadCount) +data.decode())
                Inputs = input('Server Says: ')
                if not data:
                        break
                connection.sendall(Inputs.encode())
        connection.close()
while True:
        Client, address = ServerSocket.accept()
        print('Connected to: ' + address[0] + ':' + str(address[1]))
```

```
        start_new_thread(threaded_client, (Client, ))
        ThreadCount += 1
        print('Thread Number: ' + str(ThreadCount))
ServerSocket.close()
```

client.py

```
import socket
ClientSocket = socket.socket()
host = '127.0.0.1'
port = 11596
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()
```



EXERCISE:

Q1)Write a UDP time server to display the current time and day.

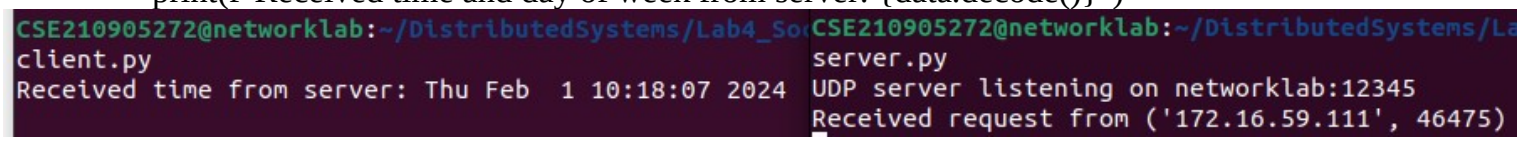
Server.py

```
import socket
import time
udp_host = socket.gethostname()# Host IP
udp_port = 12345# specified port to connect

with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
    s.bind((udp_host, udp_port))
    print(f"UDP server listening on {udp_host}:{udp_port}")

    while True:
        data, addr = s.recvfrom(1024)
        print(f"Received request from {addr}")
```

```
        now = time.ctime(time.time()) + "\r\n"

        s.sendto(now.encode(), addr)
```

client.py

```
import socket
udp_host = socket.gethostname()# Host IP
udp_port = 12345# specified port to connect

with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
    # Send a request to the server
    s.sendto(b'', (udp_host, udp_port))

    # Wait for a response from the server
    data, _ = s.recvfrom(1024)
    print(f"Received time and day of week from server: {data.decode()}")
```



Q2)Write a UDP simple chat program for message send and receive.

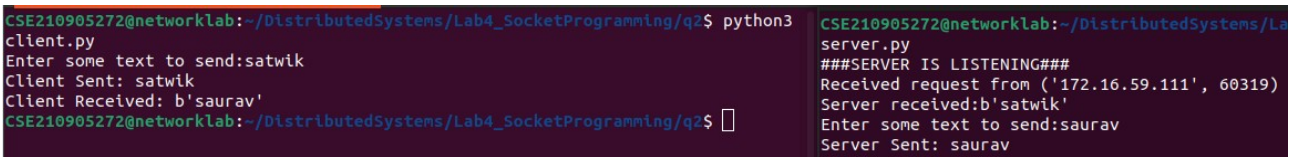
Client.py

```
import socket
udp_host = socket.gethostname()# Host IP
udp_port = 12345# specified port to connect
addr = (udp_host, udp_port)

with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
    # Send a request to the server
    s.sendto(b'', addr)

    # Wait for a response from the server
    # data, _ = s.recvfrom(1024)
    now=input("Enter some text to send:")
    s.sendto(now.encode(), addr)
    print(f"Client Sent: {now}")
    data, _ = s.recvfrom(1024)
    print(f"Client Received: {data}")
```

Server.py

```
import socket
udp_host = socket.gethostname()# Host IP
udp_port = 12345# specified port to connect

# Create a UDP socket
with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
```

```
    s.bind((udp_host, udp_port))
    print("###SERVER IS LISTENING###")

    while True:
        # Receive a request
        data, addr = s.recvfrom(1024)
        print(f"Received request from {addr}")
        data1, _ = s.recvfrom(1024)
        print(f"Server received:{data1}")
        now=input("Enter some text to send:")
        s.sendto(now.encode(), addr)
        print(f"Server Sent: {now}")
```



.

Q3)Write a TCP/UDP peer to peer chat system between two different machines.

Server.py

```
import socket
serv = ('172.16.59.52', 9991)
HOST, PORT = serv[0], serv[1]
s = socket.socket()
s.bind(serv)
s.listen()
print("waiting for incoming connections\n")
conn, addr = s.accept()
print("received connection from ", addr[0], "(", addr[1], ")\n")
s_name = conn.recv(1024).decode()
print(s_name, "has connected to the chat room")
print("enter 'bye' to exit chat room\n")
name = input(str("enter your name: "))
conn.send(name.encode())
while True:
    msg = conn.recv(1024).decode()
    if(msg=='bye'):
        conn.close()
        break
    print(s_name, ":", msg)
    msg = input(str("me: "))
    if msg == "bye":
        conn.send(msg.encode())
        print("\n")
        conn.close()
        break
```

```
    conn.send(msg.encode())
```

client.py

```
import socket
serv = ('172.16.59.52', 9991)
HOST, PORT = serv[0], serv[1]
s = socket.socket()
name = input(str("\nEnter your name: "))
print("\nTrying to connect to ", HOST, "(", PORT, ")\n")
s.connect(serv)
print("Connected...\n")
s.send(name.encode())
s_name = s.recv(1024).decode()
print(s_name, "has connected to the chat room")
print("\nEnter 'bye' to exit chat room\n")
while True:
    msg = str(input("Me : "))
    if msg == "bye":
        print("Left chat room")
        s.send(msg.encode())
        print("\n")
        s.close()
        break
    s.send(msg.encode())
    msg = s.recv(1024).decode()
    if(msg=='bye'):
        print("exit initiated by server ")
        s.close()
        break
    print(s_name, ":", msg)
```

```
CSE210905272@networklab:~/Downloads$ python3 q3_c.py

Enter your name: Satwik

Trying to connect to  172.16.59.52 ( 9991 )

Connected...

Drishaan has connected to the chat room

Enter 'bye' to exit chat room

Me : Hello
Drishaan : Hi! How are you?
Me : Lab 4 different devices
Drishaan : Connected
Me : bye
Left chat room
```

```
210905282_drishaan@networklab: ~/Documents/dsl/lab4
210905282_drishaan@networklab:~/Documents/dsl/lab4$ python3 q3_s.py
waiting for incoming connections

received connection from  172.16.59.111 ( 51070 )

Satwik has connected to the chat room
enter 'bye' to exit chat room

enter your name: Drishaan
Satwik : Hello
me: Hi! How are you?
Satwik : Lab 4 different devices
me: Connected
210905282_drishaan@networklab:~/Documents/dsl/lab4$
```

Q4)Try to debug the error in the code and execute it.

Server.py

```
import socket

serverIP = 'localhost'
serverPort = 16000
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 True:
    connection, addr = serverSock.accept()
    # data = connection.recv(1024)
    data = connection.recv(1024).decode()
    if not data: break

    temp = [float(x) for x in data.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))
    connection.send(str(msg).encode())
```

Client.py

```
import socket
serverIP = 'localhost'
serverPort = 16000
clientSock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
clientSock.connect((serverIP, serverPort))

message = input("Input integers with space in between: ").encode()
# clientSock.connect((serverIP, serverPort))
# message = raw_input("Input integers with space in between: ")
#message2 = input("Enter the length of the set: ").encode()

clientSock.send(message)
#clientSock.send(message2)

#data = clientSock.recv(1024)
data = clientSock.recv(1024).decode()
```

```
temp = [float(x) for x in data.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()
```