# **CS358: Computer Networks Lab Tutorial 3**

## **Instructions:**

# For serverX.py

- → Choose one out of four servers
- → Run the python code of the server in the terminal without any args

## Ex:- \$ python server1.py

- → User have to enter IP address and port number
- $\rightarrow$  Now server will use above information for further processes.

# For client.py

- → Choose the client python file
- → Run the python code of the server in the terminal without any args

Ex:- \$ python client.py

- → User have to enter IP address and port number
- → Now client will use above information for further processes.

#### **Instructions:**

# For serverX.py

- → server will show each step of socket forming, socket binding, listening
- → Address and port of connection will be printed whenever forms connection with a new client
- → Prints the message received over the connection
- → press 'ctrl+c' in the terminal to stop the server

# For client.py

- → client will show steps like socket forming
- → before entering message for the server, 'do you want to continue?' question will appear
- → server's reply will be printed after each message
- → to stop, user can write 'n' to discontinue.

## For client.py

Client

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60_CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
Enter port: 4000
Socket created
Want to continue?(y/n) y
-> 9-6-400
Recieved over the connection: -397
Want to continue?(y/n) y
-> abb
Recieved over the connection: Please write an expression
Want to continue?(y/n) y
-> 8/0
Recieved over the connection: Do not divide with 0
Want to continue?(y/n) y
-> 8*7-9+
Recieved over the connection: Invalid syntax
Want to continue?(y/n) y
-> 8**2
Recieved over the connection: 64
Want to continue?(y/n) n
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60_CS359/tut3$
```

## For server1.py with single client

#### Server 1

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60_CS359/tut3$ python server1.py
Enter host IP: 127.0.0.1
Enter port: 3000
Socket created
Socket binded with ip: 127.0.0.1 port: 3000
Socket is listening
connection from: 127.0.0.1 port: 56654
recieved over the connection: 4-5
recieved over the connection: 8*8
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60_CS359/tut3$
```

#### Client 1

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001C560_CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
Enter port: 3000
Socket created
Want to continue?(y/n) y
-> 4-5
Recieved over the connection: -1
Want to continue?(y/n) y
-> 8*8
Recieved over the connection: 64
Want to continue?(y/n) n
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001C560_CS359/tut3$
```

For server1.py with single client

# Client 2 trying to connect while Server 1 is connected to Client 1

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60_CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
Enter port: 3000
Socket created
Want to continue?(y/n) y
-> 6-8
Traceback (most recent call last):
   File "/home/sanskriti/Documents/GitHub/2001CS60_CS359/tut3/client.py", line 58, in <module>
        client(host_ip, port)
   File "/home/sanskriti/Documents/GitHub/2001CS60_CS359/tut3/client.py", line 33, in client
        data = client_socket.recv(1024).decode()
ConnectionResetError: [Errno 104] Connection reset by peer
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60_CS359/tut3$
```

## For server2.py with multi clients

#### Server 2

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS359/tut3$ python server2.pv
Enter host IP: 127.0.0.1
Enter port: 2000
Socket created
Socket binded with ip: 127.0.0.1 port: 2000
Socket is listening
connection from: 127.0.0.1 port: 41518
connection from: 127.0.0.1 port: 41518
Socket is listening
recieved over the connection: 5-7
connection from: 127.0.0.1 port: 40794
connection from: 127.0.0.1 port: 40794
Socket is listening
recieved over the connection: 9**2
recieved over the connection: 7-5
recieved over the connection: abc
recieved over the connection: 7/0
connection from: 127.0.0.1 port: 48174
connection from: 127.0.0.1 port: 48174
Socket is listening
recieved over the connection: 7-8-9+22
recieved over the connection: 6-9-
recieved over the connection: 99-88
```

## For server2.py with multi clients

Recieved over the connection: Invalid syntax

(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001C560\_CS359/tut3\$

Want to continue?(v/n) n

## Server 2, Client 1

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
                                                                                                                             Server 2, Client 2
Enter port: 2000
Socket created
                                                                              (base) sanskriti@sans-ubuntu:-/Documents/GitHub/2001CS60_CS359/tut3$ python client.py
Want to continue?(y/n) y
                                                                              Enter host IP: 127.0.0.1
-> 5-7
                                                                              Enter port: 2000
Recieved over the connection: -2
                                                                              Socket created
Want to continue?(y/n) y
                                                                              Want to continue?(y/n) y
-> 99-88
Recieved over the connection: 11
                                                                              Recieved over the connection: 81
                                                                              Want to continue?(y/n) y
Want to continue?(y/n)
                                                                              -> 7-5
                                                                              Recieved over the connection: 2
                                                                              Want to continue?(y/n) y
 Server 2, Client 3
                                                                              -> abc
                                                                              Recieved over the connection: Please write an expression
                                                                                       ntinue?(y/n) y
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
                                                                                       ver the connection: Do not divide with 0
Enter port: 2000
                                                                                       ntinue?(v/n)
Socket created
Want to continue?(y/n) y
-> 7-8-9+22
Recieved over the connection: 12
Want to continue?(y/n) y
-> 6-9-
```

## For server3.py with multi clients

#### Server 3

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS359/tut3$ python server3.py
Enter host IP: 127.0.0.1
Enter port: 4000
Socket created
Socket binded with ip: 127.0.0.1 port: 4000
Socket is listening
connection from: 127.0.0.1 port: 41366
recieved over the connection: 4*8-3
connection from: 127.0.0.1 port: 47822
recieved over the connection: 9-6-400
recieved over the connection: abb
recieved over the connection: 8/0
recieved over the connection: 8*7-9+
recieved over the connection: 8**2
connection from: 127.0.0.1 port: 33102
recieved over the connection: 9-8-2
recieved over the connection: 7-9-3+100
```

## For server3.py with multi clients

(base) sanskriti@sans-ubuntu:-/Documents/GitHub/2001CS60 CS359/tut3\$

## Server 3, Client 1

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
Enter port: 4000
Socket created
                                                                                                                             Server 3, Client 2
Want to continue?(y/n) y
-> 9-6-400
Recieved over the connection: -397
                                                                (base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 C5359/tut3$ python client.py
Want to continue?(y/n) y
                                                                Enter host IP: 127.0.0.1
-> abb
                                                                Enter port: 4000
Recieved over the connection: Please write an expression
                                                                Socket created
Want to continue?(v/n) v
                                                                Want to continue?(y/n) y
-> 8/0
                                                                -> 4*8-3
Recieved over the connection: Do not divide with 0
                                                                Recieved over the connection: 29
Want to continue?(y/n) y
                                                                Want to continue?(y/n) y
-> 8*7-9+
                                                                ->
Recieved over the connection: Invalid syntax
Want to continue?(y/n) y
-> 8**2
Recieved over the connection: 64
Want to continue?(y/n) n
```

## Server 3, Client 3

```
(base) sanskriti@sans-ubuntu:~/Documents/GltHub/2001C560_CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
Enter port: 2000
Socket created
Want to continue?(y/n) y
-> 7-8-9+22
Recieved over the connection: 12
Want to continue?(y/n) y
-> 6-9-
Recieved over the connection: Invalid syntax
Want to continue?(y/n) n
(base) sanskriti@sans-ubuntu:~/Documents/GltHub/2001C560_CS359/tut3$

■
```

## For server4.py with multi clients

#### Server 4

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60_CS359/tut3$ python server4.py
Enter host IP: 127.0.0.1
Enter port: 2000
Socket created
Socket binded with ip: 127.0.0.1 port: 2000
Socket is listening
connection from: 127.0.0.1 port: 45794
recieved over the connection: echo
recieved over the connection: yeah
connection from: 127.0.0.1 port: 38826
recieved over the connection: 128-99
connection from: 127.0.0.1 port: 60216
recieved over the connection: unicorn
recieved over the connection: angel
recieved over the connection: america
```

## For server4.py with multi clients

## Server 4, Client 1

```
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
Enter port: 2000
Socket created
Want to continue?(y/n) y
-> echo
Recieved over the connection: echo
Want to continue?(y/n) y
-> veah
                                                                                                               Server 4, Client 2
Recieved over the connection: yeah
Want to continue?(y/n) y
                                                             (base) sanskriti@sans-ubuntu:-/Documents/GitHub/2001CS60 CS359/tut3$ python client.py
-> america
                                                             Enter host IP: 127.0.0.1
Recieved over the connection: america
                                                             Enter port: 2000
Want to continue?(v/n) n
                                                             Socket created
(base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS3
                                                             Want to continue?(y/n) y
                                                             -> 128-99
                                                             Recieved over the connection: 128-99
 Server 4, Client 3
                                                             Want to continue?(v/n) n
                                                             (base) sanskriti@sans-ubuntu:~/Documents/GitHub/2001CS60 CS359/tut3$
```

```
(base) sanskriti@sans-ubuntu:-/Documents/GitHub/2001C560_CS359/tut3$ python client.py
Enter host IP: 127.0.0.1
Enter port: 2000
Socket created
Want to continue?(y/n) y
-> unicorn
Recieved over the connection: unicorn
Want to continue?(y/n) y
-> angel
Recieved over the connection: angel
Want to continue?(y/n)
```

## **Features:**

# For client.py

→ Warning if socket is not formed

# For server[1|2|3].py

- → Warning if socket is not formed
- → Warning if socket is not binded to the port and IP
- → Warning if port is already in use
- → Warning for invalid syntax / invalid expression / division by 0

# For server1.py

→ Can only connect to one client

#### **Features:**

# For server4.py

- → Warning if socket is not formed
- → Warning if socket is not binded to the port and IP
- → Warning if port is already in use
- → Gives the same data the user gave as it is (echo)

## **Video link:**

<u>link</u>