

NAME: RESHMA AL

DATE:19.08.25

ROLL NO.:241901089

## EXERCISE 2

### TCP CLIENT – SERVER USING SOCKET PROGRAMMING IN PYTHON

#### AIM:

To implement TCP client–server communication using socket programming in Python.

#### ALGORITHM:

##### SERVER:

1. Create a socket using `socket.socket()`.
2. Bind the socket to an IP and port using `bind ()`.
3. Listen for client connections using `listen ()`.
4. Accept client connection using `accept ()`.
5. Receive data using `recv ()`.
6. Send response using `send ()`.
7. Close connection.

##### CLIENT:

1. Create a socket using `socket.socket()`.
2. Connect to the server using `connect ()`.
3. Send data using `send ()`.
4. Receive response using `recv ()`.
5. Close connection.

#### CODE:

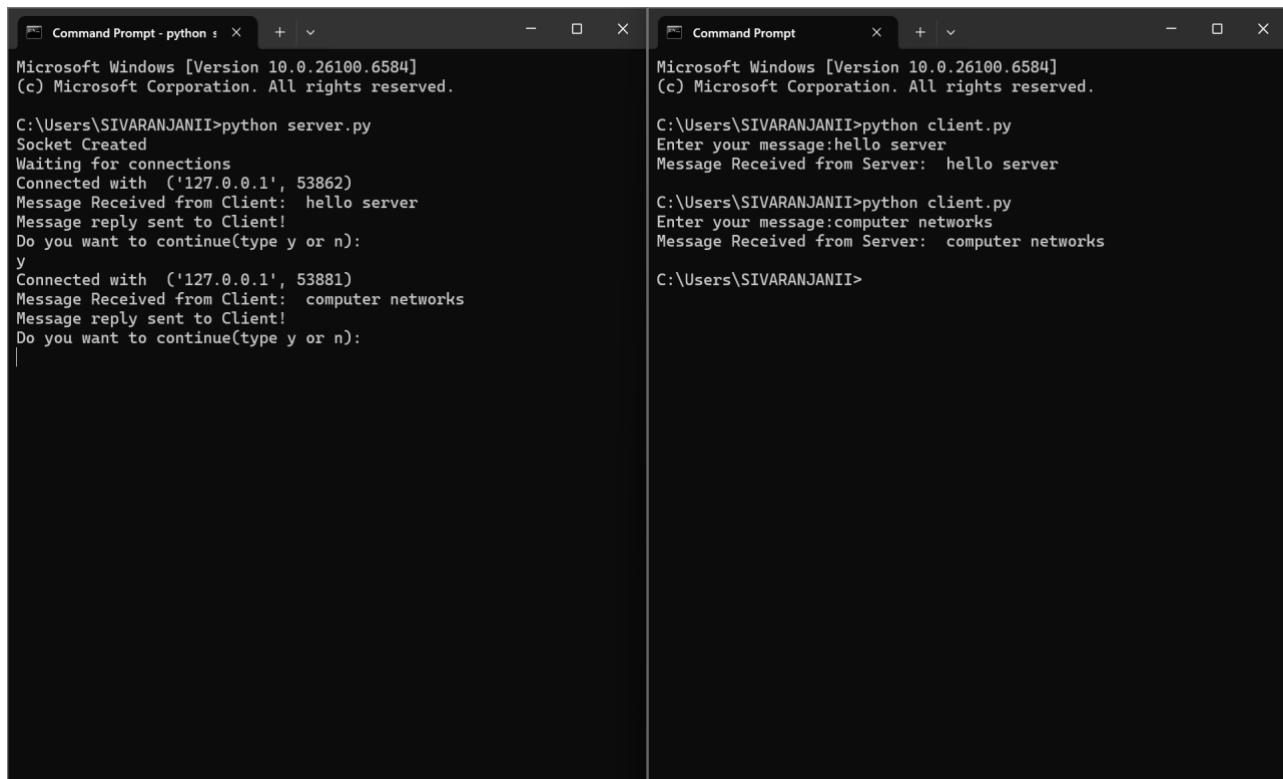
##### SERVER:

```
import socket
sockfd=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
print('Socket Created')
sockfd.bind(('localhost',55555))
sockfd.listen(3)
print('Waiting for connections')
while True:
    clientfd,addr=sockfd.accept()
    receivedMsg=clientfd.recv(1024).decode()
    print("Connected with ",addr)
    print("Message Received from Client: ",receivedMsg)
    clientfd.send(bytes(receivedMsg,'utf-8'))
    print("Message reply sent to Client!")
    print("Do you want to continue(type y or n):")
    choice=input()
    if choice=='n':
```

**CLIENT:**

```
import socket
clientfd=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
clientfd.connect(('localhost',55555))
name=input("Enter your message:")
clientfd.send(bytes(name,'utf-8'))
print("Message Received from Server: ",clientfd.recv(1024).decode())
```

**OUTPUT:**



The image shows two Microsoft Windows Command Prompt windows side-by-side. The left window, titled 'Command Prompt - python', displays the output of the server script. It starts with the Windows version information, then shows the command 'python server.py'. The server logs include: 'Socket Created', 'Waiting for connections', 'Connected with ('127.0.0.1', 53862)', 'Message Received from Client: hello server', 'Message reply sent to Client!', and a prompt 'Do you want to continue(type y or n):'. A user types 'y' and the server responds with 'Connected with ('127.0.0.1', 53881)', followed by 'Message Received from Client: computer networks' and 'Message reply sent to Client!'. The right window, also titled 'Command Prompt', displays the output of the client script. It starts with the Windows version information, then shows the command 'python client.py'. It prompts the user with 'Enter your message:' and receives 'hello server'. The client then prints 'Message Received from Server: hello server'. It then runs another instance of 'python client.py', prompting 'Enter your message:' with 'computer networks', receiving 'Message Received from Server: computer networks', and finally closing with 'C:\Users\SIVARANJANII>'.

**RESULT:**

Thus, TCP client-server communication was successfully implemented using Python.