EXPERIMENT 2 - TCP CLIENT-SERVER COMMUNICATION USING SOCKET PROGRAMMING IN PYTHON

Aim:

To study and implement socket programming in Python for establishing communication between a client and a server using the TCP/IP protocol.

Introduction:

Socket programming is a method used for communication between two programs running on a network.

It allows a client and a server to send and receive data through network connections.

Python provides a built-in module called socket that supports both TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) communication.

Algorithm:

Server:

1. Import the socket module.

2. Create a socket object using socket.socket().

3. Bind the socket to a host address and port using bind().

4. Listen for incoming connections using listen().

5. Accept a connection using accept().

6. Receive data from the client using recv().

7. Send a response to the client using send().

8. Close the connection using close().

Client:

1. Import the socket module.

2. Create a socket object using socket.socket().

3. Connect to the server using connect((host, port)).

4. Send data to the server using send().

5. Receive a response from the server using recv().

6. Close the connection using close().

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':
        break
```

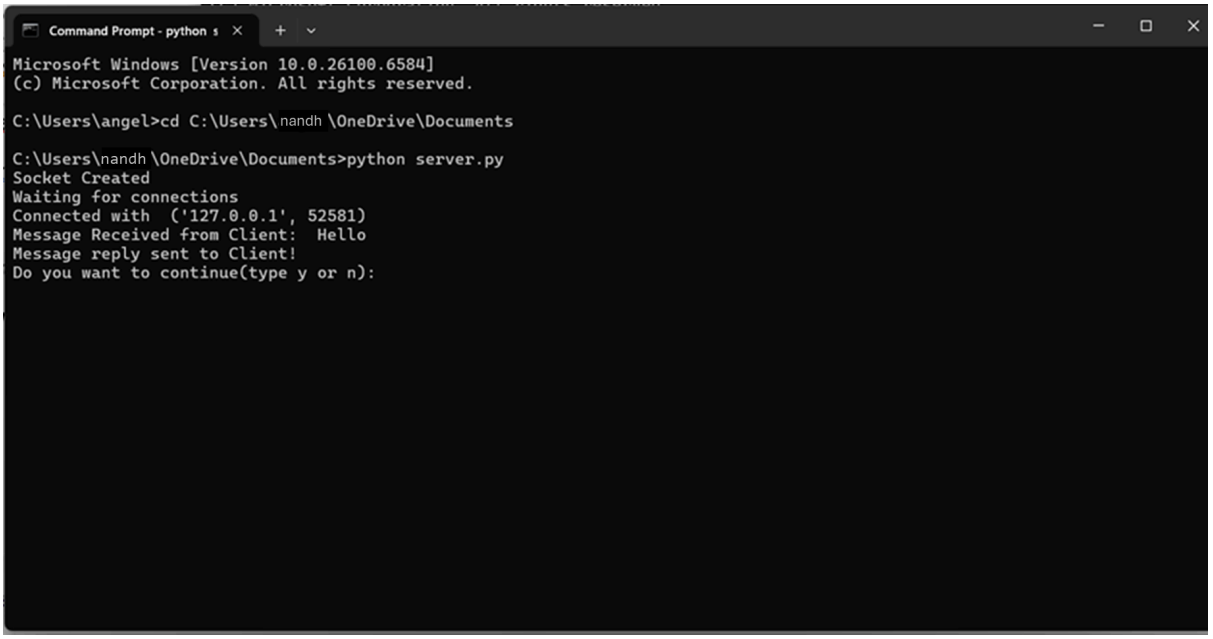
CLIENT:

```
import socket
clientfd=socket.socket(socket.AF_INET, socket.SOCK_STREAM)
clientfd.connect(('localhost',55555))
name=input("Enter your message:")
```

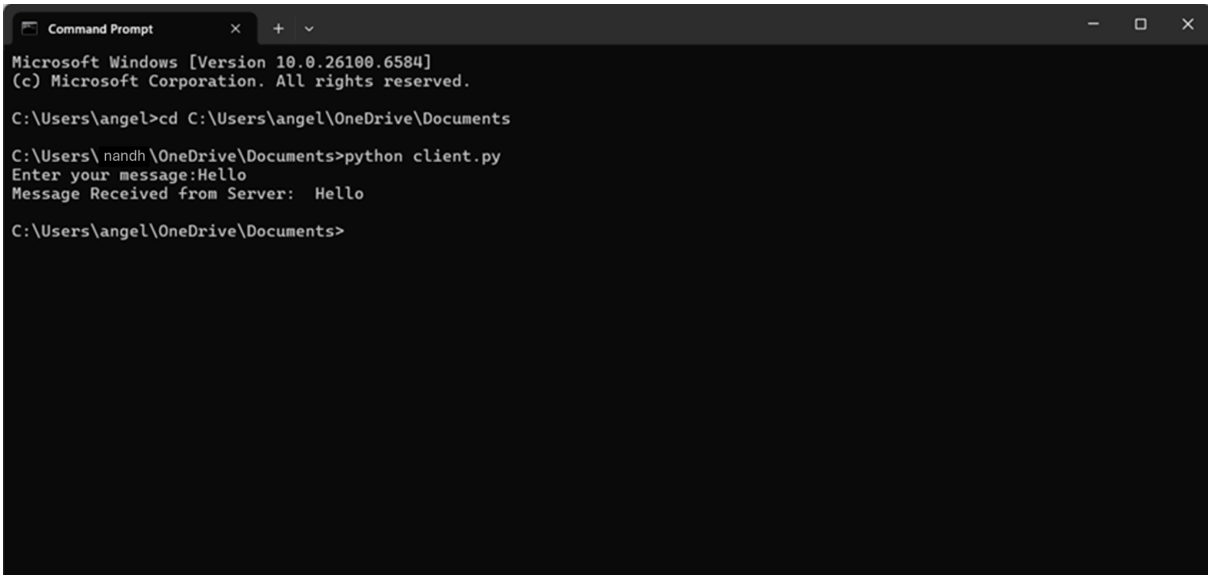
clientfd.send(bytes(name,'utf-8'))

print("Message Received from Server: ",clientfd.recv(1024).decode())

Output: Server:



Client:



Result:

The implementation of socket programming in Python was successfully executed. A TCP connection was established between the client and the server, enabling successful message exchange.