Reg.No.: 2116220701040

Practical 12

AIM:

a) Implement echo client server using TCP/UDP sockets.

ALGORITHM:

TCP Server Algorithm

1. **Initialize the Server**:
   - Create a TCP socket using socket.socket(socket.AF_INET, socket.SOCK_STREAM).
2. **Bind the Server**:
   - Bind the server socket to a specific IP address (127.0.0.1) and port (12345).
   - This will allow the server to listen for incoming connections on that IP and port.
3. **Listen for Connections**:
   - Set the server socket to listen mode using .listen().
   - This allows the server to accept multiple connections.
4. **Accept Connections in a Loop**:
   - Start an infinite loop to continuously accept client connections.
   - For each connection:
     - Use .accept() to accept the incoming connection from a client.
     - Retrieve the client's address and the socket for the connection.
5. **Handle Client Communication**:
   - Inside another loop, handle the communication with the connected client:
     - Receive data from the client using .recv(1024).

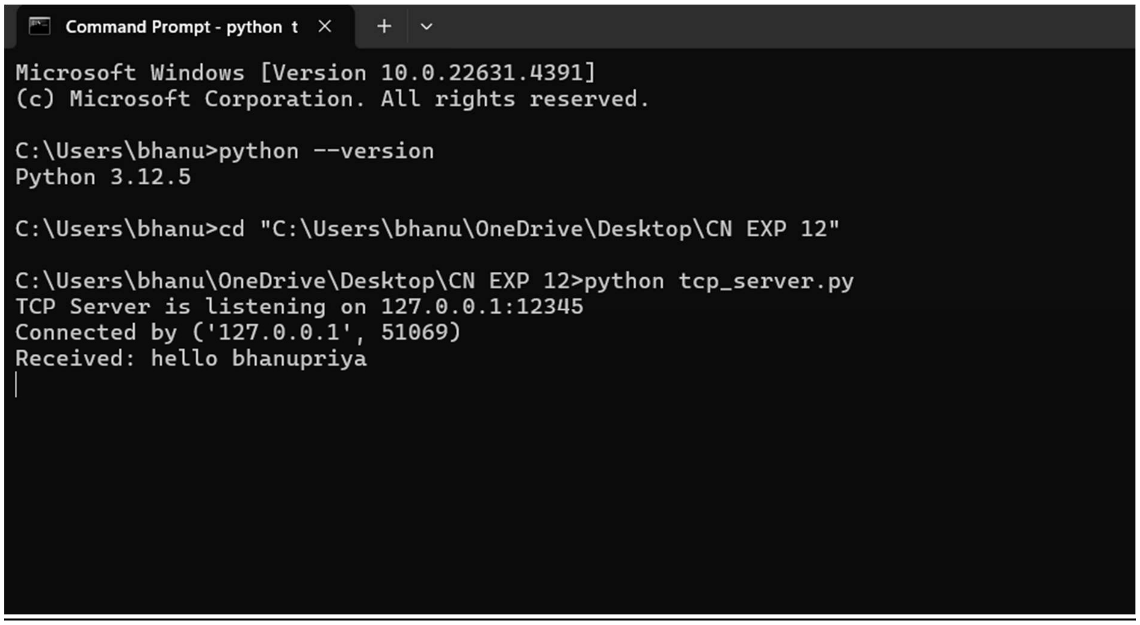
        - If no data is received, break the loop (indicating the client has disconnected).
        - Print the received data.
        - Send the received data back to the client using .sendall(data) (echo the message).

6. **Close the Connection**:
    - When the client disconnects, close the connection with that client.
    - The server continues running, ready to accept new connections.

<u>TCP Client Algorithm</u>

1. **Initialize the Client**:
    - Create a TCP socket using socket.socket(socket.AF_INET, socket.SOCK_STREAM).
2. **Connect to the Server**:
    - Connect the client socket to the server using .connect((host, port)), with host set to 127.0.0.1 and port set to 12345.
3. **Send Data to Server**:
    - Prompt the user to enter a message.
    - Encode the message and send it to the server using .sendall(message.encode()).
4. **Receive Data from Server**:
    - Wait for the server to send back data using .recv(1024).
    - Decode the received data and print it.
5. **Close the Connection**:
    - After receiving the echoed message, the client program will end, automatically closing the connection.

OUTPUT:

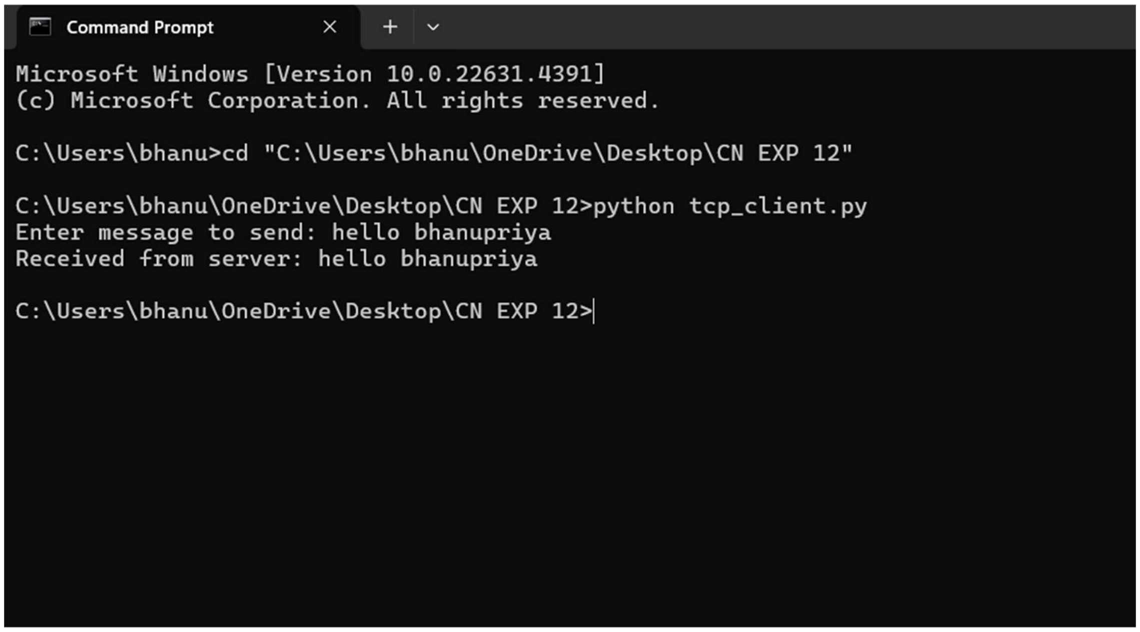


```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\bhanu>python --version
Python 3.12.5

C:\Users\bhanu>cd "C:\Users\bhanu\OneDrive\Desktop\CN EXP 12"

C:\Users\bhanu\OneDrive\Desktop\CN EXP 12>python tcp_server.py
TCP Server is listening on 127.0.0.1:12345
Connected by ('127.0.0.1', 51069)
Received: hello bhanupriya
```



```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\bhanu>cd "C:\Users\bhanu\OneDrive\Desktop\CN EXP 12"

C:\Users\bhanu\OneDrive\Desktop\CN EXP 12>python tcp_client.py
Enter message to send: hello bhanupriya
Received from server: hello bhanupriya

C:\Users\bhanu\OneDrive\Desktop\CN EXP 12>
```