Problem No: 01
Problem Name: Socket Client-Server Communication in Python

Objective:
- To implement a TCP client-server communication system using Python.
- To understand the basic concepts of socket programming and network communication.

Theory:
- **Socket:** Endpoint for sending or receiving data across a network.
- **TCP/IP:** Protocol that ensures reliable communication between two systems.
- **Client-Server Model:**
  - **Server:** Waits for connections from clients and responds.
  - **Client:** Initiates connection to send/receive data.
- **Port Number:** Identifier for a specific process/application on a machine (Port 1002 used here).

Key Components:
- Python socket library
- TCP protocol for reliable communication
- Server and client programs
- Localhost (127.0.0.1) as host

Application:
- Chat applications
- Online gaming communication
- File transfer programs
- Network services

Implementation in Python:
Server Code (server1002.py)

```
import socket
server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('localhost', 1002))
server_socket.listen(1)
print("Server is listening on port 1002...")
conn, addr = server_socket.accept()
print(f"Connected by {addr}")
while True:
    data = conn.recv(1024)
    if not data:
        break
    conn.sendall(data)
conn.close()
server_socket.close()
```

Client Code (client1002.py)

```
import socket
client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect(('localhost', 1002))
```

```
while True:
    message = input("chithi deo: ")
    if message.lower() == 'monowar':
        break
    client_socket.sendall(message.encode())
    data = client_socket.recv(1024)
    print(f"Received from server: {data.decode()}")
client_socket.close()
```

Result / Discussion:

- Client successfully connects to the server.
- Messages sent by the client are echoed back by the server.
- Communication terminates correctly when the client exits.

Sample Output:

Server Console:
Server is listening on port 1002...
Connected by ('127.0.0.1', 65129)

Client Console:
chithi deo: lab
Received from server: lab
chithi deo: monowar

Discussion:
In this lab, we implemented a TCP client-server system in Python. The client successfully connected to the server, sent messages, and received responses. This demonstrates how TCP ensures reliable communication and how the client-server model works.

Conclusion:
The lab helped us understand socket programming and the client-server architecture. Messages were transmitted and received correctly, showing successful implementation of network communication using Python.

Problem No: 02
Problem Name: Echo Socket Client-Server Communication in Python

Objective:

- To implement a socket-based client-server system that echoes messages.
- To understand how the server can process and respond to client messages.

Theory:

- **Echo Server:** A server that sends back to the client exactly what it receives.
- **TCP/IP Protocol:** Ensures reliable delivery of messages.
- **Client-Server Model:**
- **Server:** Receives messages from the client and echoes them back.
- **Client:** Sends messages and displays the server's responses.
- **Port Number:** Unique identifier for the server application use port 1002 .

Key Components:

- Python socket library
- TCP protocol
- Server and client programs
- Localhost (127.0.0.1)

Application:

- Chat systems
- Network testing tools
- Message relay services

Implementation in Python:

Server Code (server1002.py)

```
import socket

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind(('localhost', 1002))
server_socket.listen(1)
print("Echo server is listening on port 1002...")

conn, addr = server_socket.accept()
print(f"Connected by {addr}")

while True:
    data = conn.recv(1024)
    if not data:
        break
    conn.sendall(data)

conn.close()
```

```
server_socket.close()
```

Client Code (client1002.py)

```
import socket

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect(('localhost', 1002))

while True:
    message = input("chithi deo : ")
    if message.lower() == 'monowar':
        break
    client_socket.sendall(message.encode())
    data = client_socket.recv(1024)
    print(f"Echo from server: {data.decode()}")

client_socket.close()
```

Result:

The client successfully connects to the server. Messages sent from the client are echoed back by the server immediately. This demonstrates the working of an echo server and reliable TCP communication between client and server.

Sample Output:

Server Console:

```
Echo server is listening on port 1002...
Connected by ('127.0.0.1', 54321)
```

Client Console:

```
chithi deo : boss
Echo from server: boss
chithi deo : ki debo
Echo from server: ki debo
chithi deo : monowar
```

Discussion :

The lab demonstrates a simple echo server where the server sends back the same messages received from the client. TCP ensures reliable communication, and the experiment shows how client-server interaction works.

Conclusion :

This lab reinforces understanding of socket programming and echo communication. Messages are transmitted and received correctly, proving successful implementation of an echo client-server system in Python.