4. Write a program on datagram socket for client/server to display the messages on client side, typed at the server side.

File Name: UDPS.java

```
import java.io.*;
import java.net.*;

public class UDPS {
    public static void main(String[] args) {
        DatagramSocket socket = null;

        try {
            // Create a DatagramSocket at
port 6788
            socket = new
DatagramSocket(6788);
            System.out.println("Server
started. Waiting for client messages...");

            byte[] buffer = new byte[1024];

            while (true) {
                // Receive packet from
client
```

```
                DatagramPacket request = new
DatagramPacket(buffer, buffer.length);
                socket.receive(request);

                // Extract message
                String message = new
String(request.getData(), 0,
request.getLength());
                System.out.println("Received
from client: " + message);

                // Process and prepare reply
                String replyMessage =
message + " (server processed)";
                byte[] sendMsg =
replyMessage.getBytes();

                // Send reply back to client
                DatagramPacket reply = new
DatagramPacket(
                        sendMsg,
                        sendMsg.length,

request.getAddress(),
                        request.getPort()
```

```
                );

                socket.send(reply);
                System.out.println("Reply
sent to client.");
            }

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (socket != null)
                socket.close();
        }
    }
}
```

File Name: UDPC.java

```
import java.io.*;
import java.net.*;

public class UDPC {
    public static void main(String[] args) {
        DatagramSocket socket = null;
```

```
        try {
            // Create socket for client
            socket = new DatagramSocket();
            InetAddress host =
InetAddress.getByName("127.0.0.1");
            int serverPort = 6788;

            // Message to send
            String msg = "Hello Server";
            byte[] sendData =
msg.getBytes();

            // Send packet to server
            DatagramPacket request = new
DatagramPacket(sendData, sendData.length,
host, serverPort);
            socket.send(request);
            System.out.println("Message sent
to server: " + msg);

            // Receive reply
            byte[] buffer = new byte[1024];
            DatagramPacket reply = new
DatagramPacket(buffer, buffer.length);
            socket.receive(reply);
```

```
            // Display reply from server
            String received = new
String(reply.getData(), 0,
reply.getLength());
            System.out.println("Client
received: " + received);

        } catch (Exception e) {
            e.printStackTrace();
        } finally {
            if (socket != null)
                socket.close();
        }
    }
}
```

Execution Steps in VS Code:

1. Setup Your Folder

- Create a new folder, e.g., UDP.
- Inside it, create two files:
  - UDPS.java (Server)
  - UDPC.java (Client)

2. Open Folder in VS Code

- Open VS Code.
- Click **File → Open Folder → select your UDP folder**.

3. Compile Both Programs

- Open a terminal in VS Code (Ctrl + ~) and run:
  - javac UDPS.java
  - javac UDPC.java
  - This will generate UDPS.class and UDPC.class files.

4. Run the Server

- In the terminal, start the server first:
- java UDPS
- You'll see:
- Server started. Waiting for client messages...

5. Run the Client

- Open a **new terminal** (in VS Code → click "+" symbol in terminal) and run:
- java UDPC
- You'll see output like:
- Message sent to server: Hello Server
- Client received: Hello Server (server processed)
- And in the **server terminal**, you'll see:
- Received from client: Hello Server
- Reply sent to client.

6. Terminate the Programs

- Press **Ctrl + C** in each terminal to stop server and client execution.