**Lab Practical #07:**

Study Client-Server Socket programming - TCP & UDP

**Practical Assignment #07:**

1. **Write a C/Java code for TCP Server-Client Socket Programming.**
2. **Write a C/Java code for UDP Server-Client Socket Programming.**
3. **For TCP Server-Client:**

**TCP Server Program:**

import java.io.BufferedInputStream;

import java.io.DataInputStream;

import java.net.\*;

import java.io.\*;

public class Server {

    private Socket socket = null;

    private ServerSocket server = null;

    private DataInputStream in = null;

    public Server (int port) {

        //starts server and waits for a connection

        try {

            server = new ServerSocket(port);

            System.out.println("Server started...");

            System.out.println("Waiting for a client...");

            socket = server.accept();

            System.out.println("Client accepted");

            //takes input from the client socket

            in = new DataInputStream(new BufferedInputStream(socket.getInputStream()));

            String line = "";

            // reads message from client until "Over" is sent

            while(!line.equals("Over")) {

                try {

                    line = in.readUTF();

                    System.out.println(line);

                } catch(IOException e) {

                    System.out.println(e);

                }

            }

            System.out.println("Closing connection");

            //close connection

            socket.close();

            in.close();

        }

        catch(IOException e) {

            System.out.println(e);

        }

    }

    public static void main(String args[]) {

        Server server = new Server(5000);

    }

}

**TCP Client Program:**

import java.net.\*;

import java.io.\*;

public class Client1

{

    //initialize socket and i/o stream

    private Socket socket = null;

    private BufferedReader input = null;

    private DataOutputStream out = null;

    //constructor to put IP address and port

    public Client1(String address, int port) {

        //establish a connection

        try{

            socket = new Socket(address, port);

            System.out.println("Connnected");

            //take input from terminal

            input = new BufferedReader(new InputStreamReader(System.in));

            //sends output to the socket

            out = new DataOutputStream(socket.getOutputStream());

        }

        catch(UnknownHostException e) {

            System.out.println("unknownHost :: " + e);

        }

        catch(IOException e) {

            System.out.println("ioException :: " + e);

        }

        //String to read message from input tab

        String line = "";

        while(!line.equals("Over")) {

            try{

                line = input.readLine();

                out.writeUTF(line);

            }

            catch(IOException e) {

                System.out.println("ioException :: " + e);

            }

        }

        //close the connection

        try {

            input.close();

            out.close();

            socket.close();

        } catch(IOException e) {

            System.out.println("ioException :: " + e);

        }

    }

    public static void main(String args[]) {

        Client1 client = new Client1("127.0.0.1",5000);

    }

}

1. **For UDP Server-Client:**

**UDP Server Program:**

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetAddress;

**import** java.net.SocketException;

**public** **class** udpBaseServer\_2

{

**public** **static** **void** main(String[] args) **throws** IOException

    {

        // Step 1 : Create a socket to listen at port 1234

        DatagramSocket ds = **new** DatagramSocket(1234);

**byte**[] receive = **new** **byte**[65535];

        DatagramPacket DpReceive = **null**;

**while** (**true**)

        {

            // Step 2 : create a DatgramPacket to receive the data.

            DpReceive = **new** DatagramPacket(receive, receive.length);

            // Step 3 : revieve the data in byte buffer.

            ds.receive(DpReceive);

            System.out.println("Client:-" + data(receive));

            // Exit the server if the client sends "bye"

**if** (data(receive).toString().equals("bye"))

            {

                System.out.println("Client sent bye.....EXITING");

**break**;

            }

            // Clear the buffer after every message.

            receive = **new** **byte**[65535];

        }

    }

    // A utility method to convert the byte array

    // data into a string representation.

**public** **static** StringBuilder data(**byte**[] a)

    {

**if** (a == **null**)

**return** **null**;

        StringBuilder ret = **new** StringBuilder();

**int** i = 0;

**while** (a[i] != 0)

        {

            ret.append((**char**) a[i]);

            i++;

        }

**return** ret;

    }

}

**UDP Client Program:**

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetAddress;

**import** java.util.Scanner;

**public** **class** udpBaseClient\_2

{

**public** **static** **void** main(String args[]) **throws** IOException

    {

        Scanner sc = **new** Scanner(System.in);

        // Step 1:Create the socket object for

        // carrying the data.

        DatagramSocket ds = **new** DatagramSocket();

        InetAddress ip = InetAddress.getLocalHost();

**byte** buf[] = **null**;

        // loop while user not enters "bye"

**while** (**true**)

        {

            String inp = sc.nextLine();

            // convert the String input into the byte array.

            buf = inp.getBytes();

            // Step 2 : Create the datagramPacket for sending

            // the data.

            DatagramPacket DpSend =

**new** DatagramPacket(buf, buf.length, ip, 1234);

            // Step 3 : invoke the send call to actually send

            // the data.

            ds.send(DpSend);

            // break the loop if user enters "bye"

**if** (inp.equals("bye"))

**break**;

        }

    }

}