**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.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.ServerSocket;

import java.net.Socket;

public class TCPServer {

public static void main(String[] args) {

ServerSocket serverSocket = null;

Socket clientSocket = null;

BufferedReader in = null;

PrintWriter out = null;

try {

// Creating server socket

serverSocket = new ServerSocket(Integer.parseInt(args[0]));

System.out.println("Server is running on port " + args[0]);

// Accepting client connection

clientSocket = serverSocket.accept();

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

// Getting input and output streams

in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));

out = new PrintWriter(clientSocket.getOutputStream(), true);

String messageFromClient = in.readLine();

System.out.println("Received message: " + messageFromClient);

// Sending message to client

out.println("Server received your message: " + messageFromClient);

} catch (Exception e) {

e.printStackTrace();

} finally {

try {

if (in != null)

in.close();

if (out != null)

out.close();

if (clientSocket != null)

clientSocket.close();

if (serverSocket != null)

serverSocket.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

**TCP Client Program:**

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.Socket;

public class TCPClient {

public static void main(String[] args) {

Socket socket = null;

BufferedReader in = null;

PrintWriter out = null;

try{

// Creating client socket

socket = new Socket(args[0],Integer.parseInt(args[1]));

// Setting up input and output streams

in = new BufferedReader(new InputStreamReader(socket.getInputStream()));

out = new PrintWriter(socket.getOutputStream(), true);

//sending response from server

String messageFromServer = in.readLine();

System.out.println("Message fro server: " + messageFromServer);

} catch(IOException e){

e.printStackTrace();

} finally{

try {

if (in != null)

in.close();

if (out != null)

out.close();

if (socket != null)

socket.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

}

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

**UDP Server Program:**

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class UDPServer {

public static void main(String[] args) {

DatagramSocket socket = null;

try {

socket = new DatagramSocket(8080);

byte[] receiveBuffer = new byte[1024];

byte[] sendBuffer;

while (true) {

DatagramPacket receivePacket = new DatagramPacket(receiveBuffer, receiveBuffer.length);

socket.receive(receivePacket);

String clientMessage = new String(receivePacket.getData(), 0, receivePacket.getLength());

System.out.println("Client: " + clientMessage);

InetAddress clientAddress = receivePacket.getAddress();

int clientPort = receivePacket.getPort();

String serverMessage = "Hello from server";

sendBuffer = serverMessage.getBytes();

DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, clientAddress,

clientPort);

socket.send(sendPacket);

}

} catch (Exception e) {

e.printStackTrace();

} finally {

if (socket != null && !socket.isClosed()) {

socket.close();

}

}

}

}

**UDP Client Program:**

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class UDPClient {

public static void main(String[] args) {

DatagramSocket socket = null;

try {

socket = new DatagramSocket();

InetAddress serverAddress = InetAddress.getByName("localhost");

byte[] sendBuffer;

byte[] receiveBuffer = new byte[1024];

String clientMessage = "Hello from client";

sendBuffer = clientMessage.getBytes();

DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, serverAddress, 8080);

socket.send(sendPacket);

DatagramPacket receivePacket = new DatagramPacket(receiveBuffer, receiveBuffer.length);

socket.receive(receivePacket);

String serverMessage = new String(receivePacket.getData(), 0, receivePacket.getLength());

System.out.println("Server: " + serverMessage);

} catch (Exception e) {

e.printStackTrace();

} finally {

if (socket != null && !socket.isClosed()) {

socket.close();

}

}

}

}