**1.Write a Java program for TCP client-server chat (one way communication).**

**Server Code (TCPServer.java)**

import java.io.\*;

import java.net.\*;

public class TCPServer {

public static void main(String[] args) {

int port = 1234;

try (ServerSocket serverSocket = new ServerSocket(port)) {

System.out.println("Server is listening on port " + port);

Socket socket = serverSocket.accept();

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

InputStream input = socket.getInputStream();

BufferedReader reader = new BufferedReader(new InputStreamReader(input));

String message;

while ((message = reader.readLine()) != null) {

System.out.println("Client: " + message); }

socket.close();

System.out.println("Connection closed.");

} catch (IOException ex) {

System.out.println("Server exception: " + ex.getMessage());

ex.printStackTrace();

} }}

**Client Code (TCPClient.java)**

import java.io.\*;

import java.net.\*;

public class TCPClient {

public static void main(String[] args) {

String hostname = "localhost";

int port = 1234;

try (Socket socket = new Socket(hostname, port)) {

OutputStream output = socket.getOutputStream();

PrintWriter writer = new PrintWriter(output, true);

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

String text;

System.out.println("Enter messages to send to the server (type 'exit' to quit):");

while (true) {

System.out.print("You: ");

text = consoleReader.readLine();

if ("exit".equalsIgnoreCase(text)) {

break; }

writer.println(text); }

socket.close();

System.out.println("Disconnected from server.");

} catch (UnknownHostException ex) {

System.out.println("Server not found: " + ex.getMessage());

} catch (IOException ex) {

System.out.println("I/O error: " + ex.getMessage()); } }}

**2.Write a Java program for UDP server and receiver .**

**UDP Receiver (UDPServer.java)**

import java.net.\*;

public class UDPServer {

public static void main(String[] args) {

int port = 1234;

try (DatagramSocket socket = new DatagramSocket(port)) {

System.out.println("UDP Server is listening on port " + port);

byte[] buffer = new byte[1024];

while (true) {

DatagramPacket packet = new DatagramPacket(buffer, buffer.length);

socket.receive(packet);

String message = new String(packet.getData(), 0, packet.getLength());

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

if (message.equalsIgnoreCase("exit")) {

System.out.println("Server shutting down.");

break; } }

} catch (Exception ex) {

System.out.println("Server error: " + ex.getMessage());

} }}

**UDP Sender (UDPClient.java)**

import java.net.\*;

import java.io.\*;

public class UDPClient {

public static void main(String[] args) {

String hostname = "localhost";

int port = 1234;

try (DatagramSocket socket = new DatagramSocket()) {

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

String message;

System.out.println("Type messages to send to the server (type 'exit' to quit):");

while (true) {

System.out.print("You: ");

message = reader.readLine();

byte[] buffer = message.getBytes();

InetAddress address = InetAddress.getByName(hostname);

DatagramPacket packet = new DatagramPacket(buffer, buffer.length, address, port);

socket.send(packet);

if (message.equalsIgnoreCase("exit")) {

break;

} }

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

} catch (Exception ex) {

System.out.println("Client error: " + ex.getMessage());

} }}

**3.Write a Java program to get a website source code using URL .**

import java.net.\*;

import java.io.\*;

public class WebsiteSourceCode {

public static void main(String[] args) {

String websiteURL = "https://example.com";

try {

URL url = new URL(websiteURL);

URLConnection connection = url.openConnection();

BufferedReader reader = new BufferedReader(

new InputStreamReader(connection.getInputStream())

);

String line;

System.out.println("Website Source Code of " + websiteURL + ":\n");

while ((line = reader.readLine()) != null) {

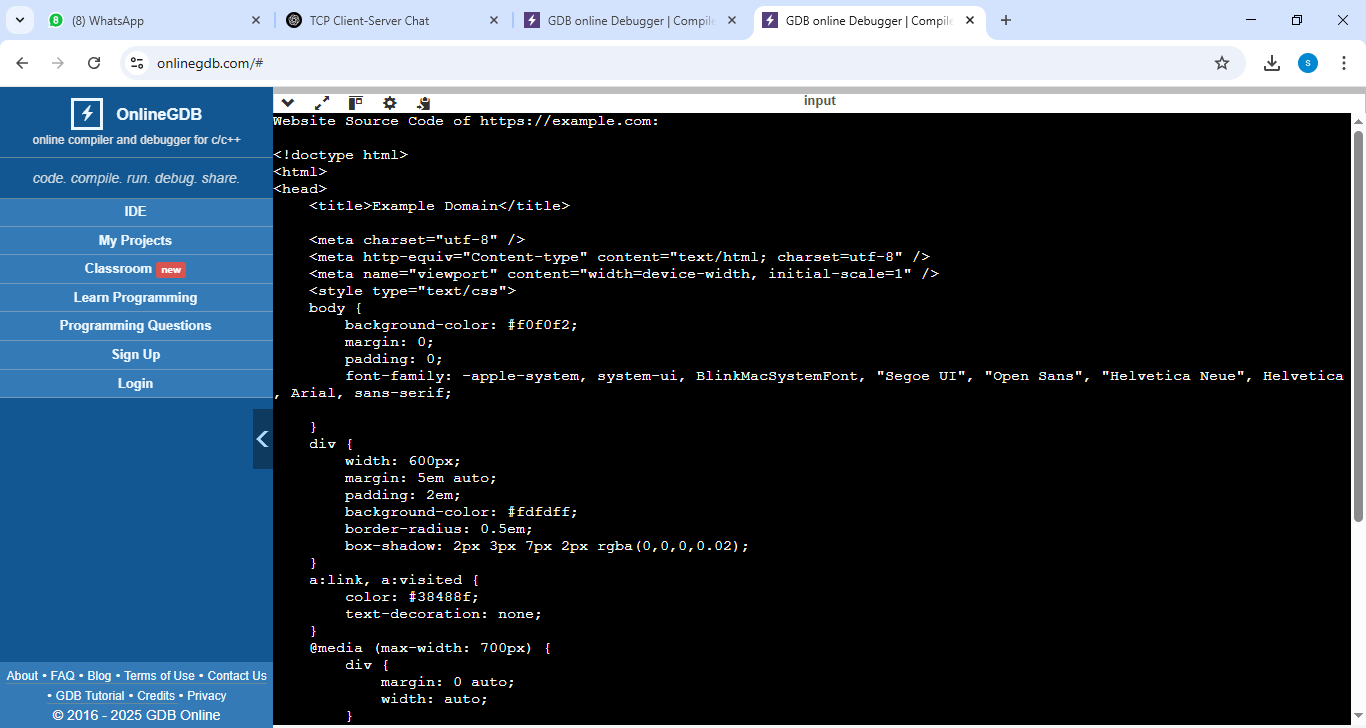
System.out.println(line); }

reader.close();

} catch (IOException e) {

System.out.println("Error fetching website source code: " + e.getMessage());

} }}



**4.Write a Java program for multi-client chat server .**

import java.io.\*;

import java.net.\*;

import java.util.\*;

public class ChatServer {

private static final int PORT = 1234;

private static Set<ClientHandler> clientHandlers = Collections.synchronizedSet(new HashSet<>());

public static void main(String[] args) {

System.out.println("Chat server started on port " + PORT + "...");

try (ServerSocket serverSocket = new ServerSocket(PORT)) {

while (true) {

Socket clientSocket = serverSocket.accept();

System.out.println("New client connected.");

ClientHandler clientHandler = new ClientHandler(clientSocket);

clientHandlers.add(clientHandler);

new Thread(clientHandler).start(); }

} catch (IOException ex) {

System.out.println("Server exception: " + ex.getMessage()); } }

static void broadcast(String message, ClientHandler sender) {

synchronized (clientHandlers) {

for (ClientHandler client : clientHandlers) {

if (client != sender) {

client.sendMessage(message);

} } }

static void removeClient(ClientHandler clientHandler) {

clientHandlers.remove(clientHandler);

}}

class ClientHandler implements Runnable {

private Socket socket;

private PrintWriter out;

private BufferedReader in;

public ClientHandler(Socket socket) {

this.socket = socket;}

public void run() {

try {

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

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

String message;

while ((message = in.readLine()) != null) {

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

ChatServer.broadcast(message, this); }

} catch (IOException e) {

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

} finally {

try {

socket.close();

} catch (IOException e) {

e.printStackTrace(); }

ChatServer.removeClient(this);} }

void sendMessage(String message) {

out.println(message); }}

**5.Write a Java program for TCP echo server.**

**TCP Echo Server (EchoServer.java)**

import java.io.\*;

import java.net.\*;

public class EchoServer {

public static void main(String[] args) {

int port = 1234;

try (ServerSocket serverSocket = new ServerSocket(port)) {

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

Socket socket = serverSocket.accept();

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

BufferedReader in = new BufferedReader(

new InputStreamReader(socket.getInputStream())

);

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

String received;

while ((received = in.readLine()) != null) {

System.out.println("Received: " + received);

out.println("Echo: " + received);

if (received.equalsIgnoreCase("exit")) {

break; } }

socket.close();

System.out.println("Connection closed.");

} catch (IOException e) {

System.out.println("Server exception: " + e.getMessage());

}}}

**TCP Echo Client (EchoClient.java)**

import java.io.\*;

import java.net.\*;

public class EchoClient {

public static void main(String[] args) {

String hostname = "localhost";

int port = 1234;

try (Socket socket = new Socket(hostname, port);

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

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

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

) {

System.out.println("Connected to Echo Server. Type messages (type 'exit' to quit):");

String inputLine;

while ((inputLine = userInput.readLine()) != null) {

out.println(inputLine);

String response = in.readLine();

System.out.println(response);

if ("exit".equalsIgnoreCase(inputLine)) {

break;

} }

socket.close();

System.out.println("Disconnected.");

} catch (IOException ex) {

System.out.println("Client exception: " + ex.getMessage());

}}}

**6.Write a Java program to display local host information .**

import java.net.\*;

public class LocalHostInfo {

public static void main(String[] args) {

try {

InetAddress localHost = InetAddress.getLocalHost();

System.out.println("Local Host Name : " + localHost.getHostName());

System.out.println("Local IP Address: " + localHost.getHostAddress());

System.out.println("Full Info : " + localHost.toString());

} catch (UnknownHostException e) {

System.out.println("Unable to retrieve local host information.");

e.printStackTrace();

}}}



**7.Write a Java program find ip address of a website.**

import java.net.\*;

public class WebsiteIPFinder {

public static void main(String[] args) {

String website = "www.google.com"; // Replace with any website

try {

InetAddress inetAddress = InetAddress.getByName(website);

System.out.println("Website: " + website);

System.out.println("IP Address: " + inetAddress.getHostAddress());

} catch (UnknownHostException e) {

System.out.println("Unable to find IP address for: " + website);

e.printStackTrace();

} }}



**8.Write a Java program to check internet connectivity.**

import java.io.IOException;

import java.net.Socket;

import java.net.UnknownHostException;

public class InternetCheck {

public static void main(String[] args) {

String host = "www.google.com";

int port = 80;

System.out.println("Checking internet connectivity...");

try (Socket socket = new Socket(host, port)) {

System.out.println("Internet is connected.");

} catch (UnknownHostException e) {

System.out.println("Host unreachable. No internet connection.");

} catch (IOException e) {

System.out.println("Cannot connect to the internet.");

} }}

