VD1

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class VD1Client {

    /\*\*

     \* @param args the command line arguments

     \*/

    public static void main (String [] args)

    {

        try{

            //Khởi tạo DatagramSocket

            DatagramSocket socket = new DatagramSocket();

            //Chuỗi tin nhắn cần gửi

            String message = "Welcome Huflit";

            //Đỉa chị ip và cổng của server

            InetAddress  serverAddress=InetAddress.getByName("127.0.0.1");

            int serverPort = 3000;

            //Chuyển đổi chuỗi tin nhắn thành mảng byte

            byte[] sendData = message.getBytes();

            //Tạo DatagramPacket để gửi dữ liệu

            DatagramPacket packet=new DatagramPacket (sendData, sendData.length,serverAddress,serverPort);

            //Gửi packet đến server

            socket.send(packet);

            //Đóng socket

            socket.close();

        } catch(IOException e){

            e.printStackTrace();

        }

    }

}

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

public class VD1Server {

    /\*\*

     \* @param args the command line arguments

    \*/

    public static void main (String [] args){

        try{

            //Khởi tạo DatagramSoket với cổng 3000

            DatagramSocket soket = new DatagramSocket(3000);

            //Mảng byte để nhận dữ liệu

            byte [] buffer = new byte [1024];

            //Tạo DatagramPacket để nhận dữ liệu

            DatagramPacket packet = new DatagramPacket(buffer,1024);

            //Nhận packet

            soket.receive(packet);

            //Chuyển đổi mảng byte nhận được thành chuỗi

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

            //In ra tin nhắn nhận được

            System.out.println(message);

            //Đóng Socket

            soket.close();

        } catch (IOException e)

        {

            e.printStackTrace();

        }

    }

}

VD 2

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class VD2Server {

    public static void main (String [] args)

    {

        try{

            int serverPort = 12345;

            //Tạo DatagramSocket để lắng nghe từ Client

            DatagramSocket serverSocket = new DatagramSocket(serverPort);

            byte [] receiveData = new byte [1024];

            while (true) {

                //Tạo DatagramPacket để nhận dữ liệu từ Client

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

                //Nhận gói tin từ Client

                serverSocket.receive(receivePacket);

                //Chuyển dữ liệu nhận được thành chuỗi

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

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

                //Phản hồi lại client

                String responseMessage= "Hello from server";

                byte [] sendData = responseMessage.getBytes();

                //Lấy địa chỉ ip và cổng client

                InetAddress clientAddress=receivePacket.getAddress();

                int clientPort = receivePacket.getPort();

                //Tạo DatagramPacket để gửi phản hồi cho Client

                DatagramPacket sendPacket= new DatagramPacket(sendData,sendData.length,clientAddress,clientPort);

                //Gửi phản hồi cho client

                serverSocket.send(sendPacket);

            }

        } catch (IOException e)

        {

            e.printStackTrace();

        }

    }

}

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class VD2Client {

    public static void main (String [] args){

        try{

            int serverPort=12345;

            String serverAddress="localhost";

            // Tạo DatagramSocket cho client

            DatagramSocket clientSocket=new DatagramSocket();

            //Dữ liệu gửi từ client đến server

            String requestData = "Hello from client";

            byte [] sendData = requestData.getBytes();

            //Tạo DatagramPacket gửi từ client đến server

            DatagramPacket sendPacket= new DatagramPacket (sendData,sendData.length,InetAddress.getByName(serverAddress),serverPort);

            //Gửi dữ liệu từ Client đến server

            clientSocket.send(sendPacket);

            byte [] receiveData = new byte[1024];

            //Tạo DatagramPacket để nhận dữ liệu từ server

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

            //Nhận dữ liệu từ server

            clientSocket.receive(receivePacket);

            //Chuyển dữ liệu nhận được thành chuỗi

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

            System.out.println("Server response: " + responseData);

            //Đóng socket của client

            clientSocket.close();

        } catch (IOException e)

        {

            e.printStackTrace();

        }

    }

}

VD 3

import java.io.ByteArrayInputStream;

import java.io.ByteArrayOutputStream;

import java.io.DataOutputStream;

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class VD3Server {

    public static void main (String [] args){

        try{

            int serverPort=12345;

            //Tạo DatagramSocket để lắng nghe kết nối từ Client

            DatagramSocket serverSocket= new DatagramSocket(serverPort);

            byte [] receiveData = new byte [1024];

            while (true) {

                // Tạo DatagramPacket để nhận dữ liệu từ client

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

                // Nhận gói tin từ Client

                serverSocket.receive(receivePacket);

                //Chuyển dữ liệu nhận được thành chuỗi

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

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

                //Phản hồi lại Client

                String responseMessage = "Hello from server";

                ByteArrayOutputStream baos = new ByteArrayOutputStream();

                DataOutputStream dos = new DataOutputStream(baos);

                dos.writeUTF(responseMessage);

                byte [] sendData = baos.toByteArray();

                //Lấy địa chỉ ip và cổng của client

                InetAddress clientAddress = receivePacket.getAddress();

                int clientPort = receivePacket.getPort();

                //Tạo DatagramPacket để gửi lại phản hồi cho client

                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,clientAddress,clientPort);

                // Gửi phản hồi cho Client

                serverSocket.send(sendPacket);

            }

        }catch (IOException e)

        {

            e.printStackTrace();

        }

    }

}

import java.io.ByteArrayInputStream;

import java.io.DataInputStream;

import java.io.EOFException;

import java.io.IOException;

import java.io.UTFDataFormatException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import javax.imageio.IIOException;

public class VD3Client {

        public static void main (String[] args)

        {

            try{

                int serverPort=12345;

                String serverAddress="localhost";

                //Tạo Datagram Socket cho client

                DatagramSocket clientSocket= new DatagramSocket();

                // Dữ liệu gửi từ client đến server

                String requestData = "Hello from Client";

                byte [] sendData=requestData.getBytes();

                //Tạo DatagramPacket để gửi từ client đến server

                DatagramPacket sendPacket= new DatagramPacket(sendData, sendData.length,InetAddress.getByName(serverAddress),serverPort);

                //Gửi dữ liệu từ client đến server

                clientSocket.send(sendPacket);

                byte [] receiveData = new byte[1024];

                //Tạo DatagramPacket để nhận dữ liệu từ server

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

                //Nhận dữ liệu từ server

                clientSocket.receive(receivePacket);

                //Chuyển dữ liệu nhận được thành chuỗi sử dụng DataInputStream

                DataInputStream din = new DataInputStream(new ByteArrayInputStream(receivePacket.getData()));

                String receivedMessage = din.readUTF();

                System.out.println("Server response: " + receivedMessage);

                //Đóng socket của client

                clientSocket.close();

            } catch(EOFException e)

            {

                e.printStackTrace();

            } catch (UTFDataFormatException e)

            {

                e.printStackTrace();

            } catch (IOException e)

            {

                e.printStackTrace();

            }

        }

}

Bài 1

import java.net.DatagramPacket;

import java.net.DatagramSocket;

public class Bai1Server {

    public static void main(String[] args) {

        DatagramSocket socket = null;

        try {

            socket = new DatagramSocket(12345);

            byte[] receiveBuffer = new byte[1024];

            System.out.println("Server is listening...");

            while (true) {

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

                socket.receive(receivePacket);

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

                int number = Integer.parseInt(receivedData.trim());

                int square = number \* number;

                String squareStr = Integer.toString(square);

                byte[] sendBuffer = squareStr.getBytes();

                DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, receivePacket.getAddress(), receivePacket.getPort());

                socket.send(sendPacket);

                System.out.println("Received number " + number + " from " + receivePacket.getAddress() + ":" + receivePacket.getPort() + ", sent back " + square);

            }

        } catch (Exception e) {

            e.printStackTrace();

        } finally {

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

                socket.close();

            }

        }

    }

}

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.util.Scanner;

public class Bai1Client {

    public static void main(String[] args) {

        DatagramSocket socket = null;

        try {

            socket = new DatagramSocket();

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

            Scanner scanner = new Scanner(System.in);

            System.out.print("Enter a number: ");

            int number = scanner.nextInt();

            String numberStr = Integer.toString(number);

            byte[] sendBuffer = numberStr.getBytes();

            DatagramPacket sendPacket = new DatagramPacket(sendBuffer, sendBuffer.length, serverAddress, 12345);

            socket.send(sendPacket);

            byte[] receiveBuffer = new byte[1024];

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

            socket.receive(receivePacket);

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

            System.out.println("The square of " + number + " is " + receivedData);

        } catch (Exception e) {

            e.printStackTrace();

        } finally {

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

                socket.close();

            }

        }

    }

}

A screenshot of a computer program

Description automatically generated

Bài 2

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class Bai2Server {

    public static void main(String[] args) {

        DatagramSocket socket = null;

        try {

            // Tạo socket với cổng 9876

            socket = new DatagramSocket(9876);

            byte[] receiveData = new byte[1024];

            byte[] sendData;

            System.out.println("Server is running...");

            while (true) {

                // Nhận dữ liệu từ client

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

                socket.receive(receivePacket);

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

                // Tính tổng dãy số

                String[] numArray = numbers.split(",");

                int sum = 0;

                for (String num : numArray) {

                    sum += Integer.parseInt(num.trim());

                }

                // Gửi kết quả lại cho client

                String result = String.valueOf(sum);

                sendData = result.getBytes();

                InetAddress clientAddress = receivePacket.getAddress();

                int clientPort = receivePacket.getPort();

                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, clientAddress, clientPort);

                socket.send(sendPacket);

            }

        } catch (Exception e) {

            e.printStackTrace();

        } finally {

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

                socket.close();

            }

        }

    }

}

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.util.Scanner;

public class Bai2Client {

    public static void main(String[] args) {

        DatagramSocket socket = null;

        try {

            socket = new DatagramSocket();

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

            byte[] sendData;

            byte[] receiveData = new byte[1024];

            Scanner scanner = new Scanner(System.in);

            System.out.print("Enter a series of numbers (comma-separated, max 50 numbers): ");

            String input = scanner.nextLine();

            // Gửi dãy số tới server

            sendData = input.getBytes();

            DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, serverAddress, 9876);

            socket.send(sendPacket);

            // Nhận kết quả từ server

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

            socket.receive(receivePacket);

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

            // In kết quả ra màn hình

            System.out.println("Sum from server: " + result);

        } catch (Exception e) {

            e.printStackTrace();

        } finally {

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

                socket.close();

            }

        }

    }

}

A screenshot of a computer program

Description automatically generated

Bài 3

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class Bai3Server {

    public static void main(String[] args) {

        DatagramSocket socket = null;

        try {

            socket = new DatagramSocket(12345);

            byte[] buffer = new byte[1024];

            System.out.println("Server is listening on port 12345...");

            while (true) {

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

                socket.receive(request);

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

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

                String reversedMessage = new StringBuilder(message).reverse().toString();

                System.out.println("Sending reversed message: " + reversedMessage);

                byte[] responseBuffer = reversedMessage.getBytes();

                DatagramPacket response = new DatagramPacket(responseBuffer, responseBuffer.length, request.getAddress(), request.getPort());

                socket.send(response);

            }

        } catch (Exception e) {

            e.printStackTrace();

        } finally {

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

                socket.close();

            }

        }

    }

}

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.util.Scanner;

public class Bai3Client {

    public static void main(String[] args) {

        DatagramSocket socket = null;

        Scanner scanner = new Scanner(System.in);

        try {

            socket = new DatagramSocket();

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

            int serverPort = 12345;

            System.out.print("Enter a string to send to the server: ");

            String message = scanner.nextLine();

            byte[] buffer = message.getBytes();

            DatagramPacket request = new DatagramPacket(buffer, buffer.length, serverAddress, serverPort);

            socket.send(request);

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

            byte[] responseBuffer = new byte[1024];

            DatagramPacket response = new DatagramPacket(responseBuffer, responseBuffer.length);

            socket.receive(response);

            String reversedMessage = new String(response.getData(), 0, response.getLength());

            System.out.println("Received reversed message from server: " + reversedMessage);

        } catch (Exception e) {

            e.printStackTrace();

        } finally {

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

                socket.close();

            }

            scanner.close();

        }

    }

}

A screenshot of a computer

Description automatically generated