VD1

Server

import java .io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class VD1Server {

    public static void main(String[] args) {

        try{

            DatagramSocket socket = new DatagramSocket();

            InetAddress broadcastAddress = InetAddress.getByName("255.255.255.255");

            String message = "This is a broadcast message ";

            byte [] sendData = message.getBytes();

            DatagramPacket sendPacket = new DatagramPacket(sendData,sendData.length,broadcastAddress,8888);

            socket.send(sendPacket);

            System.out.println("Broadcast message sent");

            socket.close();

        }catch (IOException e){

            e.printStackTrace();

        }

    }

}

Client

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

public class VD1Client {

    public static void main (String [] args ){

        try{

            DatagramSocket socket = new DatagramSocket(8888);

            byte [] receiveData = new byte[1024];

            System.out.println("Waiting for broadcast message... ");

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

            socket.receive(receivePacket);

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

            InetAddress senderAddress= receivePacket.getAddress();

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

            socket.close();

        }catch(IOException e)

        {

            e.printStackTrace();

        }

    }

}

Kết quả

A screenshot of a computer program

Description automatically generated

VD2

Server

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{

            DatagramSocket socket = new DatagramSocket();

            InetAddress broadcastAddress = InetAddress.getByName("230.0.0.0");

            String message = "This is a multicast message ";

            byte[]sendData= message.getBytes();

            DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,broadcastAddress,8888);

            socket.send(sendPacket);

            System.out.println("Multicast message sent");

            socket.close();

        }catch (IOException e)

        {

            e.printStackTrace();

        }

    }

}

Client 1

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.InetSocketAddress;

import java.net.MulticastSocket;

import java.net.NetworkInterface;

public class VD2Client1 {

    public static void main (String [] args)

    {

        try{

            InetAddress group = InetAddress.getByName("230.0.0.0");

            int port = 8888;

            MulticastSocket socket = new MulticastSocket(port);

            NetworkInterface netIf = NetworkInterface.getByName("yourNetworkInterfaceName");

            socket.joinGroup(new InetSocketAddress(group, port),netIf);

            byte[] receiveData=new byte[1024];

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

            socket.receive(receivePacket);

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

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

            socket.leaveGroup(new InetSocketAddress(group, port), netIf);

            socket.close();

        }catch(IOException e)

        {

            e.printStackTrace();

        }

    }

}

Client 2

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.InetSocketAddress;

import java.net.MulticastSocket;

import java.net.NetworkInterface;

public class VD2Client2 {

    public static void main(String[] args) {

        try{

            InetAddress group = InetAddress.getByName("230.0.0.0");

            int port= 8888;

            MulticastSocket socket = new MulticastSocket(port);

            NetworkInterface netIf = NetworkInterface.getByName("yourNetworkInterfaceName");

            socket.joinGroup(new InetSocketAddress(group, port),netIf);

            byte[] receiveData= new byte[1024];

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

            socket.receive(receivePacket);

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

            System.out.println("Received mesage: " + receivedMessage);

            socket.leaveGroup(new InetSocketAddress(group, port), netIf);

            socket.close();

        }catch(IOException e)

        {

            e.printStackTrace();

        }

    }

}

Kết quả

A screenshot of a computer

Description automatically generated

Bài 1

Server

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Bai1Server {

    public static void main(String[] args) {

        try {

            DatagramSocket socket = new DatagramSocket();

            InetAddress broadcastAddress = InetAddress.getByName("255.255.255.255");

            while (true) {

                // Get the current time

                String currentTime = new SimpleDateFormat("HH:mm:ss").format(new Date());

                byte[] sendData = currentTime.getBytes();

                // Create a packet with the current time to broadcast

                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, broadcastAddress, 8888);

                socket.send(sendPacket);

                System.out.println("Broadcast message sent: " + currentTime);

                // Wait for 1 second before sending the next time

                Thread.sleep(1000);

            }

        } catch (IOException | InterruptedException e) {

            e.printStackTrace();

        }

    }

}

Client

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

public class Bai1Client {

    public static void main(String[] args) {

        try {

            DatagramSocket socket = new DatagramSocket(8888);

            while (true) {

                // Buffer to receive incoming data

                byte[] receiveData = new byte[1024];

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

                // Receive the packet

                socket.receive(receivePacket);

                // Convert the received data to a string

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

                System.out.println("Received time: " + currentTime);

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

Kết quả

A screenshot of a computer

Description automatically generated

Bài 2

Server

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.DatagramSocket;

import java.net.InetAddress;

import java.text.SimpleDateFormat;

import java.util.Date;

public class Bai2Server {

    public static void main(String[] args) {

        try {

            DatagramSocket socket = new DatagramSocket();

            InetAddress multicastAddress = InetAddress.getByName("230.0.0.0");

            while (true) {

                // Get the current time

                String currentTime = new SimpleDateFormat("HH:mm:ss").format(new Date());

                byte[] sendData = currentTime.getBytes();

                // Create a packet with the current time to multicast

                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, multicastAddress, 8888);

                socket.send(sendPacket);

                System.out.println("Multicast message sent: " + currentTime);

                // Wait for 1 second before sending the next time

                Thread.sleep(1000);

            }

        } catch (IOException | InterruptedException e) {

            e.printStackTrace();

        }

    }

}

Client

import java.io.IOException;

import java.net.DatagramPacket;

import java.net.InetAddress;

import java.net.MulticastSocket;

public class Bai2Client {

    public static void main(String[] args) {

        try {

            MulticastSocket socket = new MulticastSocket(8888);

            InetAddress group = InetAddress.getByName("230.0.0.0");

            socket.joinGroup(group);

            while (true) {

                byte[] buffer = new byte[256];

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

                socket.receive(packet);

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

                System.out.println("Received time: " + receivedTime);

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

Kết quả

A screen shot of a computer

Description automatically generated

Bài 3

Server

import java.io.\*;

import java.net.\*;

public class Bai3Server {

    public static void main(String[] args) {

        int BROADCAST\_PORT = 8888;

        int RIDE\_PORT = 9999;

        try {

            DatagramSocket socket = new DatagramSocket(BROADCAST\_PORT);

            System.out.println("Driver is waiting for ride requests...");

            while (true) {

                byte[] buffer = new byte[256];

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

                socket.receive(packet);

                String message = new String(packet.getData(), 0, packet.getLength(), "UTF-8");

                if (message.equals("request ride")) {

                    InetAddress clientAddress = packet.getAddress();

                    int clientPort = packet.getPort();

                    System.out.println("Received ride request from client at " + clientAddress);

                    // Simulate connecting to the client and starting the ride

                    new Thread(() -> handleRide(clientAddress, RIDE\_PORT)).start();

                }

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

    private static void handleRide(InetAddress clientAddress, int ridePort) {

        try (Socket rideSocket = new Socket(clientAddress, ridePort);

             PrintWriter out = new PrintWriter(new OutputStreamWriter(rideSocket.getOutputStream(), "UTF-8"), true);

             BufferedReader in = new BufferedReader(new InputStreamReader(rideSocket.getInputStream(), "UTF-8"))) {

            // Simulate ride duration

            long startTime = System.currentTimeMillis();

            Thread.sleep(15000); // Simulate 15 seconds ride

            long endTime = System.currentTimeMillis();

            // Calculate fare

            long rideTime = (endTime - startTime) / 1000;

            int fare;

            if (rideTime <= 10) {

                fare = (int) rideTime \* 800;

            } else {

                fare = 10 \* 800 + (int) (rideTime - 10) \* 950;

            }

            // Send end of ride message with fare and total time

            String endMessage = String.format("Ride ended. Total fare: %d VND, total time: %d seconds", fare, rideTime);

            out.println(endMessage);

            System.out.println("Ride ended. Sent fare: " + fare + " VND, total time: " + rideTime + " seconds");

        } catch (IOException | InterruptedException e) {

            e.printStackTrace();

        }

    }

}

Client

import java.io.\*;

import java.net.\*;

public class Bai3Client {

    public static void main(String[] args) {

        int BROADCAST\_PORT = 8888;

        int RIDE\_PORT = 9999;

        try {

            DatagramSocket socket = new DatagramSocket();

            InetAddress broadcastAddress = InetAddress.getByName("255.255.255.255");

            String requestMessage = "request ride";

            byte[] sendData = requestMessage.getBytes("UTF-8");

            DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, broadcastAddress, BROADCAST\_PORT);

            socket.send(sendPacket);

            System.out.println("Broadcast ride request sent");

            // Listen for a driver response

            try (ServerSocket serverSocket = new ServerSocket(RIDE\_PORT);

                 Socket driverSocket = serverSocket.accept();

                 BufferedReader in = new BufferedReader(new InputStreamReader(driverSocket.getInputStream(), "UTF-8"))) {

                System.out.println("Connected to driver. Starting ride...");

                // Simulate ride duration

                long startTime = System.currentTimeMillis();

                Thread.sleep(15000); // Simulate 15 seconds ride

                long endTime = System.currentTimeMillis();

                // Read end of ride message

                String endMessage = in.readLine();

                System.out.println("Received from driver: " + endMessage);

                // Calculate and display fare

                long rideTime = (endTime - startTime) / 1000;

                int fare;

                if (rideTime <= 10) {

                    fare = (int) rideTime \* 800;

                } else {

                    fare = 10 \* 800 + (int) (rideTime - 10) \* 950;

                }

                System.out.println("Total fare: " + fare + " VND");

            } catch (IOException | InterruptedException e) {

                e.printStackTrace();

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

Kết quả

A screenshot of a computer

Description automatically generated