Buổi 2

Vd 1

import java.io.IOException;

import java.net.Socket;

import java.net.UnknownHostException;

public class VD1 {

    public static void main(String [] args){

        String host ="localhost";

        //Quet cac cong tu 1 den 1024

        for (int port=1; port <= 1024; port++){

            try{

                Socket socket =new Socket(host,port);

                System.out.println("port"+port+"is open");

                socket.close();

            }

            catch(UnknownHostException e ){

                System.out.println("The Server adress is unknown");

                break;

            }

            catch(IOException e){

                System.out.println("The Server is not found");

            }

        }

    }

}

VD2

Server

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.net.ServerSocket;

import java.net.Socket;

public class VD2 {

    public static void main (String [] args){

        try{

            // Tạo một ServerSorket để lắng nghe kết nối từ máy khách

            ServerSocket serverSocket = new ServerSocket(12345);

            System.out.println("Server is running and waiting for client connection...");

            //Chấp nhận kết nối từ máy khách

            Socket clienSocket = serverSocket.accept();

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

            //Lấy InputStream và OutputStream từ Socket để gừi và nhận dữ liệu

            InputStream inputStream = clienSocket.getInputStream();

            OutputStream outputStream =clienSocket.getOutputStream();

            //Đọc dữ liệu từ máy khách

            byte [] buffer =new byte[1024];

            int bytesRead = inputStream.read(buffer);

            String clientMessage = new String (buffer,0,bytesRead);

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

            //Gửi phản hồi lạo cho máy tính

            String serverMessage="Hello from server!";

            outputStream.write(serverMessage.getBytes());

            // Đóng kết nối

            clienSocket.close();

            serverSocket.close();

        } catch(IOException e){

            e.printStackTrace();

        }

    }

}

Client

import java.io.IOException;

import java.io.InputStream;

import java.io.OutputStream;

import java.net.Socket;

public class Clientvd2 {

    public static void main (String [] args)

    {

        try{

            // Tạo một Socket kết nối tới máy chủ

            Socket socket=new Socket("localhost",12345);

            //Lấy InputStream và OutputStream từ Socket để gừi và nhận dữ liệu

            InputStream inputStream = socket.getInputStream();

            OutputStream outputStream= socket.getOutputStream();

            //Gửi tin nhắn đến máy chủ

            String message = "Hello from client";

            outputStream.write(message.getBytes());

            //Đọc phản hồi từ máy chủ

            byte [] buffer= new byte[1024];

            int bytesRead = inputStream.read(buffer);

            String serverResponse = new String(buffer,0,bytesRead);

            System.out.println("Server respone: " + serverResponse);

            //Đóng kết nối

            socket.close();

        }catch(IOException e)

        {

            e.printStackTrace();

        }

    }

}

A screenshot of a computer program

Description automatically generated

VD3

Server

import java.io.\*;

import java.net.\*;

public class VD3 {

    public static void main (String [] args)

    {

        try{

            //Tạo một ServerSocket và ràng buộc nó với cổng 1234

            ServerSocket serverSocket = new ServerSocket(1234);

            System.out.println("Server is running and waiting for client connections...");

            //Chấp nhận yêu cầu kết nối từ một máy tính

            Socket clientSocket = serverSocket.accept();

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

            //Mở luồng vào/ra để giao tiếp với máy tính

            DataOutputStream dataOutputStream = new DataOutputStream(clientSocket.getOutputStream());

            DataInputStream dataInputStream = new DataInputStream(clientSocket.getInputStream());

            //Đọc dữ liệu từ máy khách và gửi phàn hồi

            String receiveData = dataInputStream.readUTF();

            System.out.println("Client says: " + receiveData);

            dataOutputStream.writeUTF("Server received your message: " + receiveData);

            dataOutputStream.flush();

            //Đóng kết nối

            clientSocket.close();

            serverSocket.close();

        } catch (IOException e)

        {

            e.printStackTrace();

        }

    }

}

Client

import java.io.\*;

import java.net.\*;

public class Clientvd3 {

    public static void main (String [] agrs)

    {

        try{

            //Tạo một kết nối Socket tới địa chỉ IP và cổng của máy chủ

            Socket socket = new Socket("localhost",1234);

            //Mở luồng vào/ra để giao tiếp với máy chủ

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

            DataInputStream dataInputStream = new DataInputStream(socket.getInputStream());

            //Gửi thông điệp tới máy chủ

            String message ="Hello Server!!!!";

            dataOutputStream.writeUTF(message);

            dataOutputStream.flush();

            //Đọc phản hồi từ máy chủ

            String receiveLine = dataInputStream.readUTF();

            System.out.println(receiveLine);

            //Đóng kết nối

            socket.close();

        } catch (IOException e)

        {

            e.printStackTrace();

        }

    }

}

A screenshot of a computer program

Description automatically generated

Bài 1

Server

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.OutputStreamWriter;

import java.io.PrintWriter;

import java.net.ServerSocket;

import java.net.Socket;

public class Bai1Server {

    public static void main(String[] args) {

        try (ServerSocket serverSocket = new ServerSocket(12345)) {

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

            while (true) {

                Socket socket = serverSocket.accept();

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

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

                PrintWriter writer = new PrintWriter(new OutputStreamWriter(socket.getOutputStream()), true);

                String receivedString = reader.readLine();

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

                String reversedString = new StringBuilder(receivedString).reverse().toString();

                writer.println(reversedString);

                System.out.println("Sent to client: " + reversedString);

                socket.close();

            }

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

}

Client

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.OutputStreamWriter;

import java.io.PrintWriter;

import java.net.Socket;

import java.util.Scanner;

public class Bai1Client {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter server address: ");

        String localhost = scanner.nextLine();

        try (Socket socket = new Socket("localhost", 12345);

             PrintWriter writer = new PrintWriter(new OutputStreamWriter(socket.getOutputStream()), true);

             BufferedReader reader = new BufferedReader(new InputStreamReader(socket.getInputStream()))) {

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

            String inputString = scanner.nextLine();

            writer.println(inputString);

            String reversedString = reader.readLine();

            System.out.println("Reversed string from server: " + reversedString);

        } catch (Exception e) {

            e.printStackTrace();

        }

        scanner.close();

    }

}

A screenshot of a computer program

Description automatically generated

Bài2

Server

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.ServerSocket;

import java.net.Socket;

import java.util.Random;

public class Bai2Server {

    public static void main(String[] args) {

        try {

            ServerSocket serverSocket = new ServerSocket(1234);

            System.out.println("Server is running and waiting for client connection...");

            while (true) {

                Socket clientSocket = serverSocket.accept();

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

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

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

                // Đọc số N từ client

                int N = Integer.parseInt(inputFromClient.readLine());

                // Thực hiện tính toán Pi bằng phương pháp Monte Carlo

                double pi = calculatePi(N);

                // Gửi kết quả Pi về client

                outputToClient.println(pi);

                // Đóng kết nối với client

                clientSocket.close();

            }

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

    // Hàm tính số Pi bằng phương pháp Monte Carlo

    private static double calculatePi(int N) {

        Random rand = new Random();

        int circlePoints = 0;

        int squarePoints = 0;

        for (int i = 0; i < N; i++) {

            double x = rand.nextDouble();

            double y = rand.nextDouble();

            double distance = x \* x + y \* y;

            if (distance <= 1) {

                circlePoints++;

            }

            squarePoints++;

        }

        return 4.0 \* circlePoints / squarePoints;

    }

}

Client

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.Socket;

public class Bai2Client {

    public static void main(String[] args) {

        try {

            Socket socket = new Socket("localhost", 1234);

            System.out.println("Connected to server.");

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

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

            // Gửi số N đến server

            int N = 1000000; // Số điểm để tính Pi

            outputToServer.println(N);

            // Nhận kết quả Pi từ server

            double pi = Double.parseDouble(inputFromServer.readLine());

            System.out.println("Estimated value of Pi: " + pi);

            // Đóng kết nối

            socket.close();

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

}

A screenshot of a computer program

Description automatically generated

Bài 3

Server

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.ServerSocket;

import java.net.Socket;

import java.net.URL;

import java.net.URLConnection;

public class Bai3Server {

    public static void main(String[] args) {

        try {

            ServerSocket serverSocket = new ServerSocket(1234);

            System.out.println("Server is running and waiting for client connection...");

            while (true) {

                Socket clientSocket = serverSocket.accept();

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

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

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

                // Đọc địa chỉ IP từ client

                String ipAddress = inputFromClient.readLine();

                // Kiểm tra định dạng của địa chỉ IP

                if (isValidIPAddress(ipAddress)) {

                    // Gửi thông tin về địa chỉ IP về client

                    String ipInfo = getIPInfo(ipAddress);

                    outputToClient.println(ipInfo);

                } else {

                    outputToClient.println("Invalid IP address format");

                }

                // Đóng kết nối với client

                clientSocket.close();

            }

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

    // Hàm kiểm tra định dạng của địa chỉ IP

    private static boolean isValidIPAddress(String ipAddress) {

        // Kiểm tra xem địa chỉ IP có hợp lệ hay không

        String ipPattern =

            "^((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\\.){3}" +

            "(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$";

        return ipAddress.matches(ipPattern);

    }

    // Hàm tra cứu thông tin địa chỉ IP từ ip-api.com

    private static String getIPInfo(String ipAddress) {

        try {

            // Construct the URL

            String apiUrl = "http://ip-api.com/json/" + ipAddress;

            System.out.println("API URL: " + apiUrl); // Debug output

            // Open a connection to the URL

            URL url = new URL(apiUrl);

            URLConnection connection = url.openConnection();

            // Read the response

            BufferedReader reader = new BufferedReader(new InputStreamReader(connection.getInputStream()));

            StringBuilder response = new StringBuilder();

            String line;

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

                response.append(line);

            }

            reader.close();

            return response.toString();

        } catch (IOException e) {

            e.printStackTrace(); // Print the exception stack trace for debugging

            return "Error retrieving IP information";

        }

    }

}

Client

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.Socket;

public class Bai3CLient {

    public static void main(String[] args) {

        try {

            Socket socket = new Socket("localhost", 1234);

            System.out.println("Connected to server.");

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

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

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

            // Nhập địa chỉ IP từ người dùng

            System.out.print("Enter public IP address: ");

            String ipAddress = inputFromUser.readLine();

            // Gửi địa chỉ IP đến server

            outputToServer.println(ipAddress);

            // Nhận và in ra thông tin từ server

            String response = inputFromServer.readLine();

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

            // Đóng kết nối

            socket.close();

        } catch (IOException e) {

            e.printStackTrace();

        }

    }

}

A screenshot of a computer program

Description automatically generated