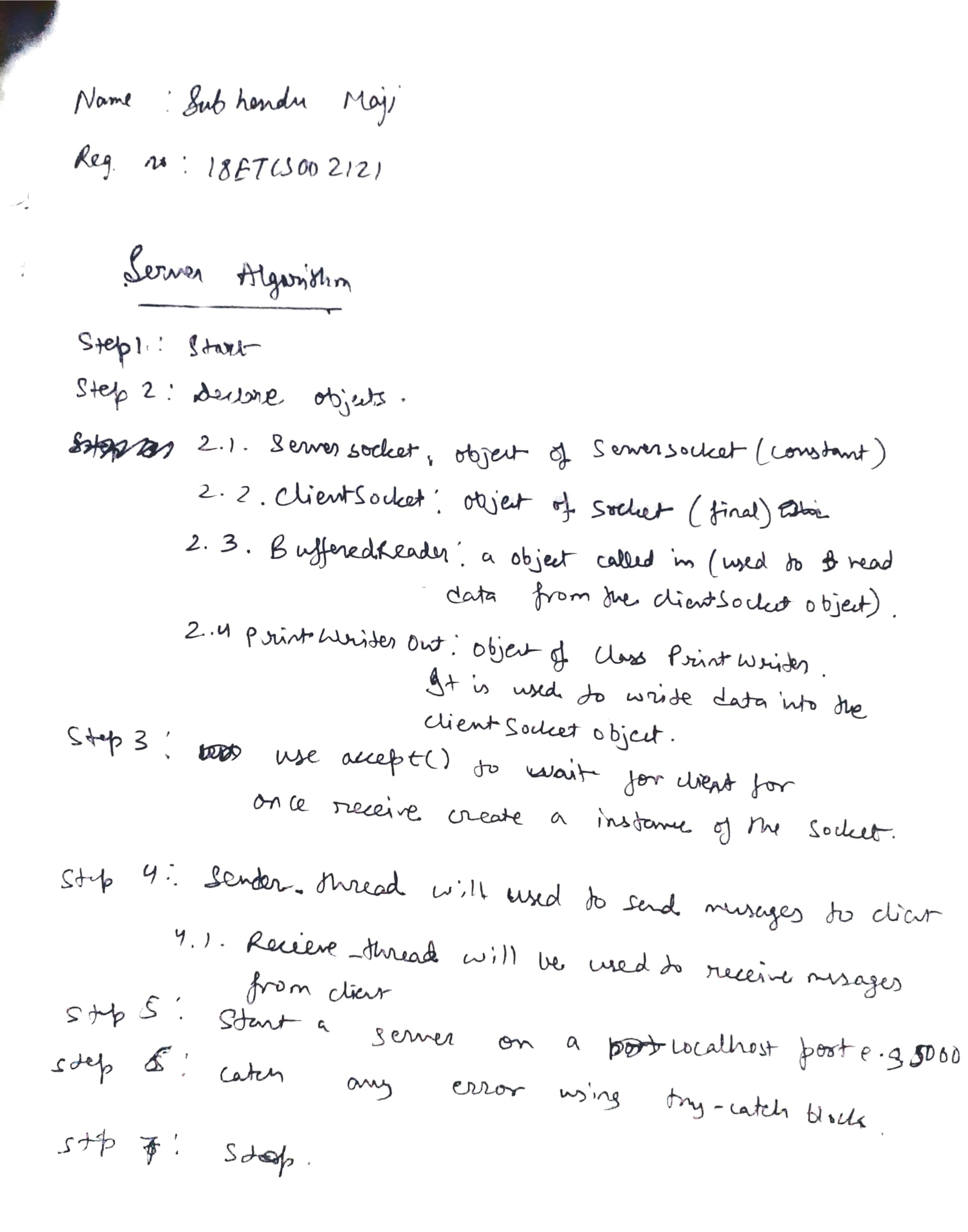
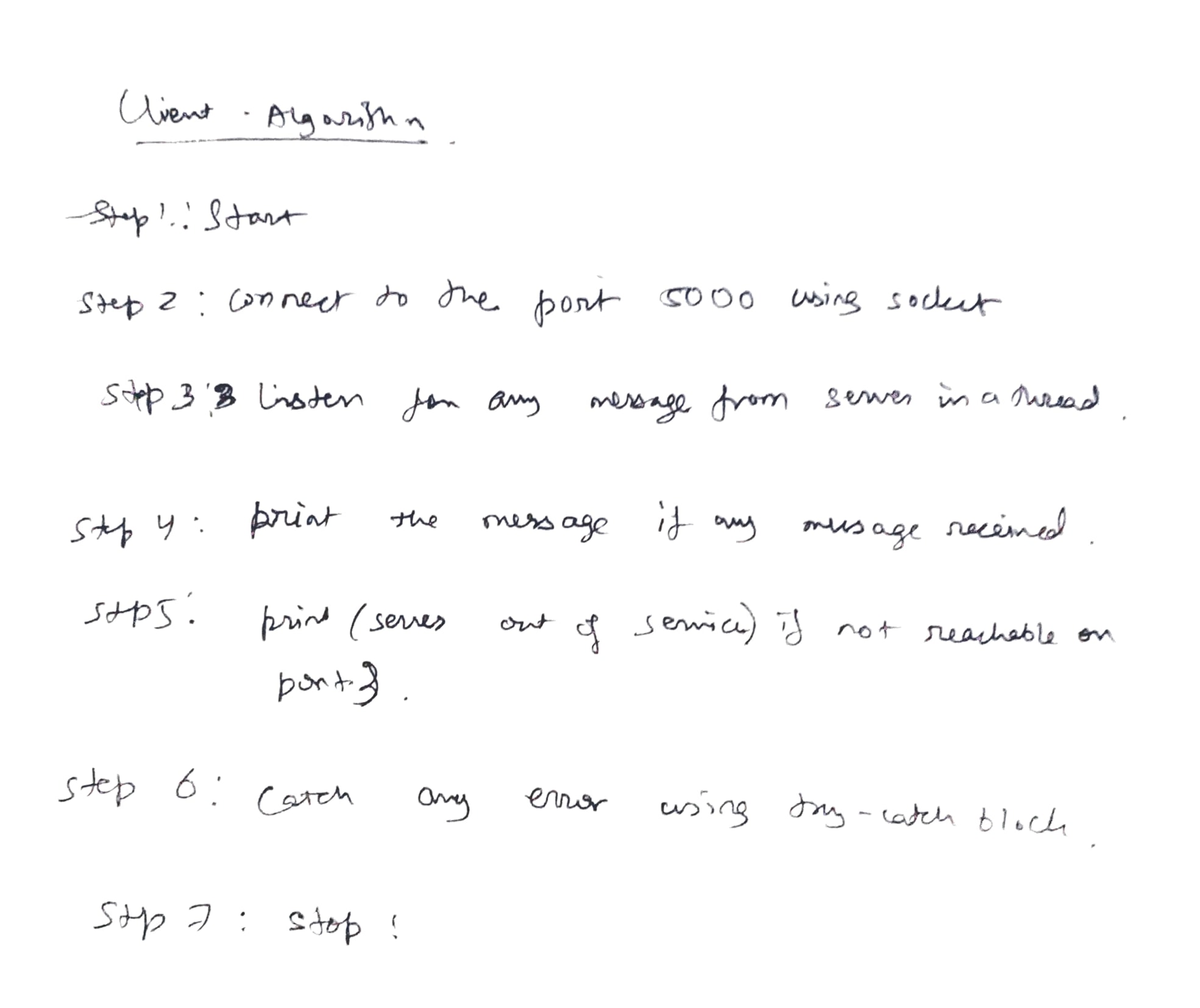
Question :Chat - Application – communication between Server and Client

**Algorithm**

Algorithm-Server



Algorithm - Client



**Source Code**

Server Code

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.util.Scanner;

public class Server {

    public static void main(String[] *args*) {

        final ServerSocket serverSocket;

        final Socket clientSocket;

        final BufferedReader in;

        final PrintWriter out;

        final Scanner sc = new Scanner(System.in);

        try {

            serverSocket = new ServerSocket(5000);

            clientSocket = serverSocket.accept();

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

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

            Thread sender = new Thread(new Runnable() {

                String msg; *// variable that will contains the data writter by the user*

                @Override *// annotation to override the run method*

                public void run() {

                    while (true) {

                        msg = sc.nextLine(); *// reads data from user's keybord*

                        out.println(msg); *// write data stored in msg in the clientSocket*

                        out.flush(); *// forces the sending of the data*

                    }

                }

            });

            sender.start();

            Thread receive = new Thread(new Runnable() {

                String msg;

                @Override

                public void run() {

                    try {

                        msg = in.readLine();

*// tant que le client est connecté*

                        while (msg != null) {

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

                            msg = in.readLine();

                        }

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

                        out.close();

                        clientSocket.close();

                        serverSocket.close();

                    } catch (IOException *e*) {

                        e.printStackTrace();

                    }

                }

            });

            receive.start();

        } catch (IOException *e*) {

            e.printStackTrace();

        }

    }}

Client Code

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.PrintWriter;

import java.net.Socket;

import java.util.Scanner;

public class Client {

    public static void main(String[] *args*) {

        final Socket clientSocket;*// socket used by client to send and recieve data from server*

        final BufferedReader in; *// object to read data from socket*

        final PrintWriter out; *// object to write data into socket*

        final Scanner sc = new Scanner(System.in); *// object to read data from user's keybord*

        try {

            clientSocket = new Socket("127.0.0.1", 5000);

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

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

            Thread sender = new Thread(new Runnable() {

                String msg;

                @Override

                public void run() {

                    while (true) {

                        msg = sc.nextLine();

                        out.println(msg);

                        out.flush();

                    }

                }

            });

            sender.start();

            Thread receiver = new Thread(new Runnable() {

                String msg;

                @Override

                public void run() {

                    try {

                        msg = in.readLine();

                        while (msg != null) {

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

                            msg = in.readLine();

                        }

                        System.out.println("Server out of service");

                        out.close();

                        clientSocket.close();

                    } catch (IOException *e*) {

                        e.printStackTrace();

                    }

                }

            });

            receiver.start();

        } catch (IOException *e*) {

            e.printStackTrace();

        }

    }

}

Output:

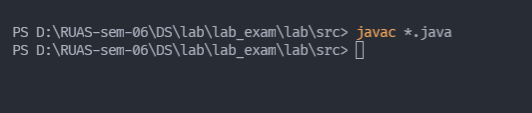


Figure creating class files that can be run on JVM

Test 1:

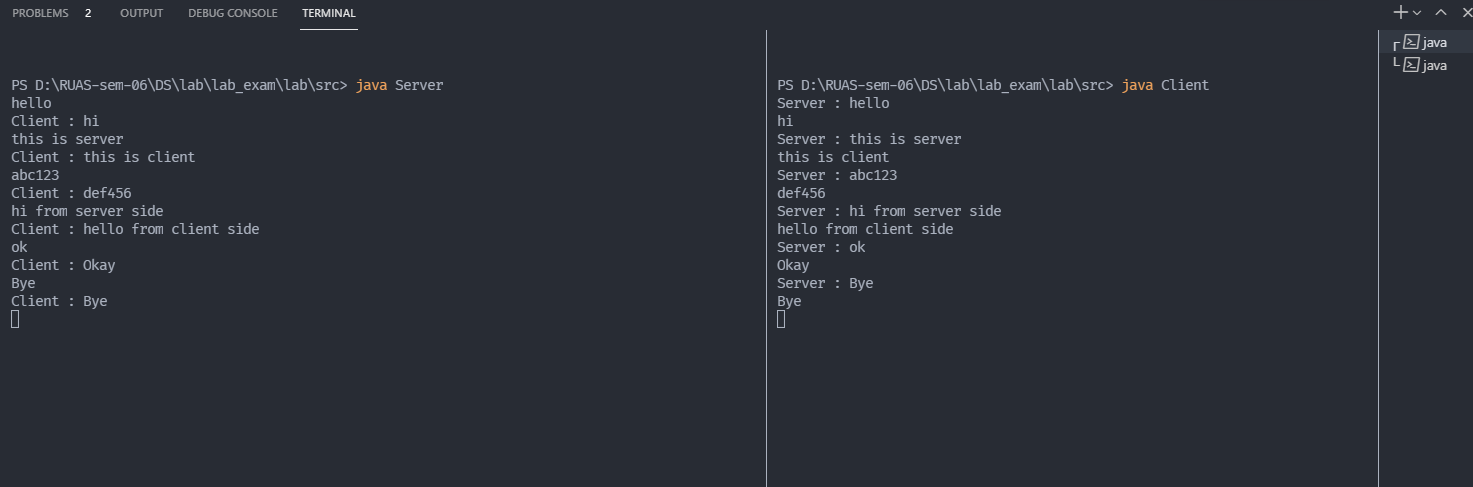


Figure Conversation between client and server

Test 2:



Figure Conversation between client and server