/\*\*\* Assignment UDP chat multiuser client \*\*\*/

import java.net.\*;

import java.io.\*;

public class Client {

public static void main(String[] args) {

Socket clientSocket = null;

DataInputStream is = null;

PrintStream os = null;

DataInputStream inputLine = null;

/\*

\* Open a socket on port 2222. Open the input and the output streams.

\*/

try {

clientSocket = new Socket("localhost", 2222);

os = new PrintStream(clientSocket.getOutputStream());

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

inputLine = new DataInputStream(new BufferedInputStream(System.in));

} catch (UnknownHostException e) {

System.err.println("Don't know about host");

} catch (IOException e) {

System.err.println("Couldn't get I/O for the connection to host");

}

/\*

\* If everything has been initialized then we want to write some data to the

\* socket we have opened a connection to on port 2222.

\*/

if (clientSocket != null && os != null && is != null) {

try {

/\*

\* Keep on reading from/to the socket till we receive the "Ok" from the

\* server, once we received that then we break.

\*/

System.out.println("The client started. Type any text. To quit it type 'Ok'.");

String responseLine;

os.println(inputLine.readLine());

while ((responseLine = is.readLine()) != null) {

System.out.println(responseLine);

if (responseLine.indexOf("Ok") != -1) {

break;

}

os.println(inputLine.readLine());

}

/\*

\* Close the output stream, close the input stream, close the socket.

\*/

os.close();

is.close();

clientSocket.close();

} catch (UnknownHostException e) {

System.err.println("Trying to connect to unknown host: " + e);

} catch (IOException e) {

System.err.println("IOException: " + e);

}

}

}

}

/\*\*\* Assignment UDP chat multiuser server \*\*\*/

import java.io.\*;

import java.net.\*;

public class Server {

public static void main(String args[]) {

ServerSocket echoServer = null;

String line;

DataInputStream is;

PrintStream os;

Socket clientSocket = null;

/\*

\* Open a server socket on port 2222. Note that we can't choose a port less

\* than 1023 if we are not privileged users (root).

\*/

try {

echoServer = new ServerSocket(2222);

} catch (IOException e) {

System.out.println(e);

}

/\*

\* Create a socket object from the ServerSocket to listen to and accept

\* connections. Open input and output streams.

\*/

System.out.println("The server started. To stop it press <CTRL><C>.");

try {

clientSocket = echoServer.accept();

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

os = new PrintStream(clientSocket.getOutputStream());

/\* As long as we receive data, echo that data back to the client. \*/

while (true) {

line = is.readLine();

os.println("From server: " + line);

}

} catch (IOException e) {

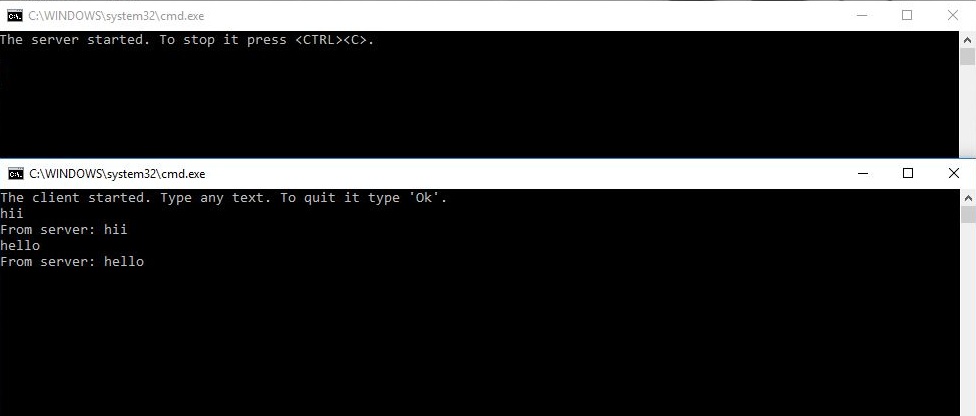
System.out.println(e);

}

}

}

\*\*\* OUTPUT of UDP Multiuser chat \*\*\*



/\*\*\* Assignment UDP chat p2p client \*\*\*/

import java.io.\*;

import java.net.\*;

class UDPClient

{

public static void main(String args[]) throws Exception

{

BufferedReader inFromUser =

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

DatagramSocket clientSocket = new DatagramSocket();

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

byte[] sendData = new byte[1024];

byte[] receiveData = new byte[1024];

String sentence = inFromUser.readLine();

sendData = sentence.getBytes();

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

clientSocket.send(sendPacket);

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

clientSocket.receive(receivePacket);

String modifiedSentence = new String(receivePacket.getData());

System.out.println("FROM SERVER:" + modifiedSentence);

clientSocket.close();

}

}

/\*\*\* Assignment UDP chat p2p server \*\*\*/

import java.io.\*;

import java.net.\*;

class UDPServer

{

public static void main(String args[]) throws Exception

{

DatagramSocket serverSocket = new DatagramSocket(9876);

byte[] receiveData = new byte[1024];

byte[] sendData = new byte[1024];

while(true)

{

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

serverSocket.receive(receivePacket);

String sentence = new String( receivePacket.getData());

System.out.println("RECEIVED: " + sentence);

InetAddress IPAddress = receivePacket.getAddress();

int port = receivePacket.getPort();

String capitalizedSentence = sentence.toUpperCase();

sendData = capitalizedSentence.getBytes();

DatagramPacket sendPacket =

new DatagramPacket(sendData, sendData.length, IPAddress, port);

serverSocket.send(sendPacket);

}

}

\*\*\* OUTPUT of UDP p2p\*\*\*

