A.S.	IIS G. B. Pentasuglia	VOTO
2017-2018	5 B informatica - Matera *Laboratorio di:* **TPS, Sistemi** *Professori:* **COSOLA, SPECCHIA**	10/10
	Linguaggi utilizzati: **Java, SQL**	

Dati dello studente	
MORELLI SAVERIO	**17/02/2018**
✓ Lavoro personale	× Lavoro di gruppo

MOME

Consegna/Traccia esercitazione

Un'azienda poduttrice di energia elettrica gestisce i contatori elettronici (smart meter) installati presso i suoi clienti.
I consumi vengono rilevati ogni 15 minuti.
La trasmissione avviene tramite un modulo chiamato MOME costituito da un microprocessore, un modem power-line e due interfacce usb.
Tutti i contatori sono in rete e hanno un proprio indirizzo IP ed impiegano per la comunicazione il protocollo di trasporto UDP.
Rispondono con un datagram di 3 Bytes contenente la lettura dei consumi (che si incrementano in modo casuale) quando il datagram di richiesta contiene il codice ASCII del carattere "?".

- Realizzare in linguaggio Java un programma di monitoraggio che interroghi i contatori ogni 15 minuti salvando in un database le risposte ricevute (l'indirizzo IP di provenienza, la data e l'ora).
- Realizzare anche il modello E/R.

Database - Tabelle:

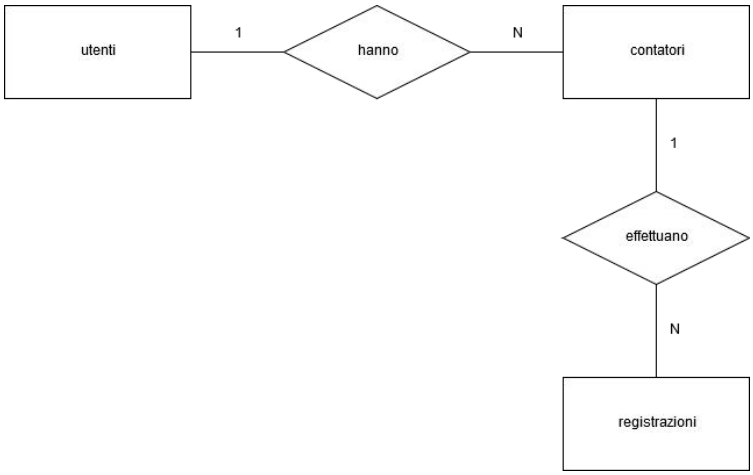
1- registrazioni (id AI PK, contatore_ip FK, data, consumo)
2- contatori (ip_address PK, nome_utente FK, ultima_rilevazione)
3- utenti (id AI PK, nome, cognome)

Analisi esercitazione

Per prima cosa è necessario creare la tabella contenente gli utenti proprietari dei contatori mome, quindi la tabella **utenti** ed ogni utente può avere più contatori intestati, quindi si ha un'altra tabella denominata **contatori**.
In utenti si hanno solamente i dati della persona, mentre in contatori si hanno i dati relativi a quel determinato contatore, quali l'indirizzo IP, l'ultima registrazione ed il codice identificativo dell'utente (proprietario).
Dopodichè è necessario creare la tabella contenente tutte le registrazioni, quindi **registrazioni** che avrà come campi un id identificativo, l'id identificativo del contatore (ovvero l'indirizzo IP, il quale è univoco), la data della rilevazione e il consumo totale della rilevazione).

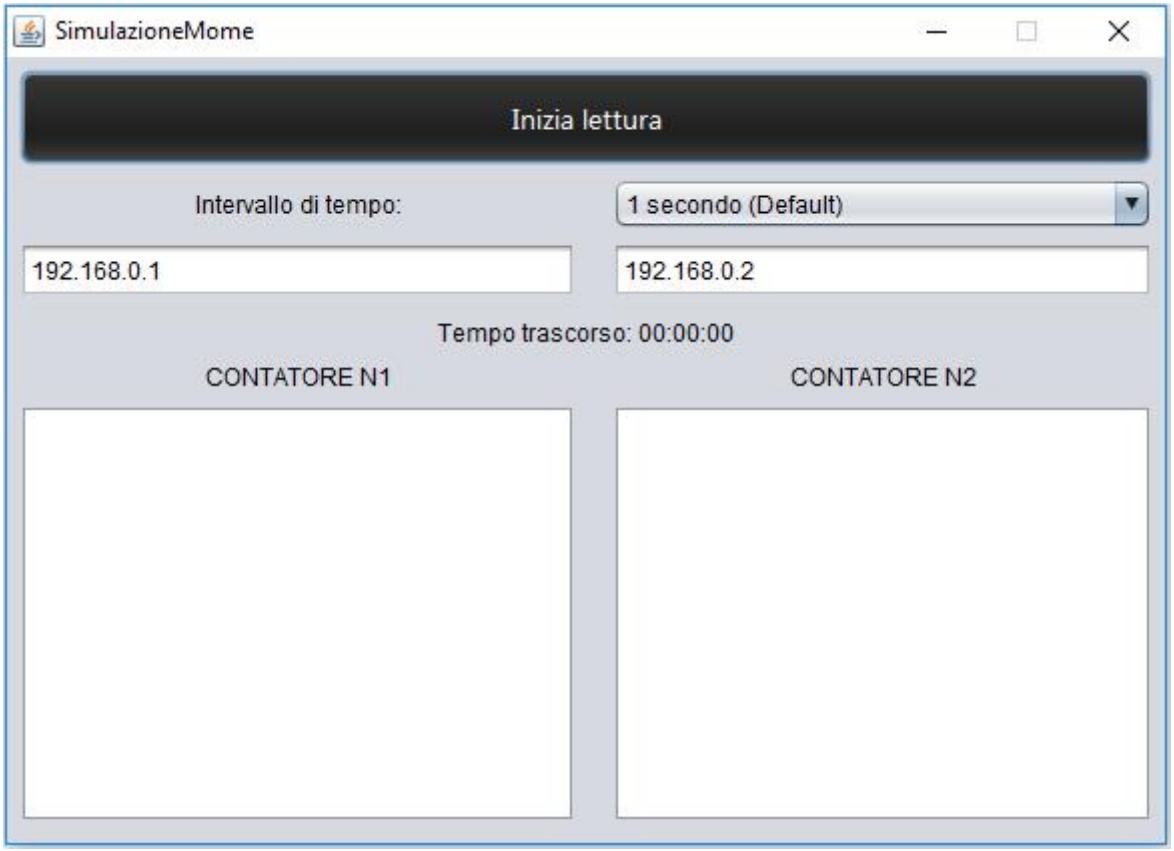
Modello E/R (generato automaticamente da phpMyAdmin)

Morello E/R - Concettuale



Modello E/R - Logico

Per rendere il tutto più facile all'utente è stata creata un'interfaccia grafica (GUI) ed inoltre non è necessario avviare manualmente il server perchè il simulatore, non appena si preme "Inizia lettura" lo avvia in automatico (e lo chiude quando si preme "Termina lettura").



Nell'esempio sono stati considerati solamente 2 utenti (e ciascuno ha un singolo contatore), tuttavia tutto ciò funzionerebbe anche con più utenti e contatori.

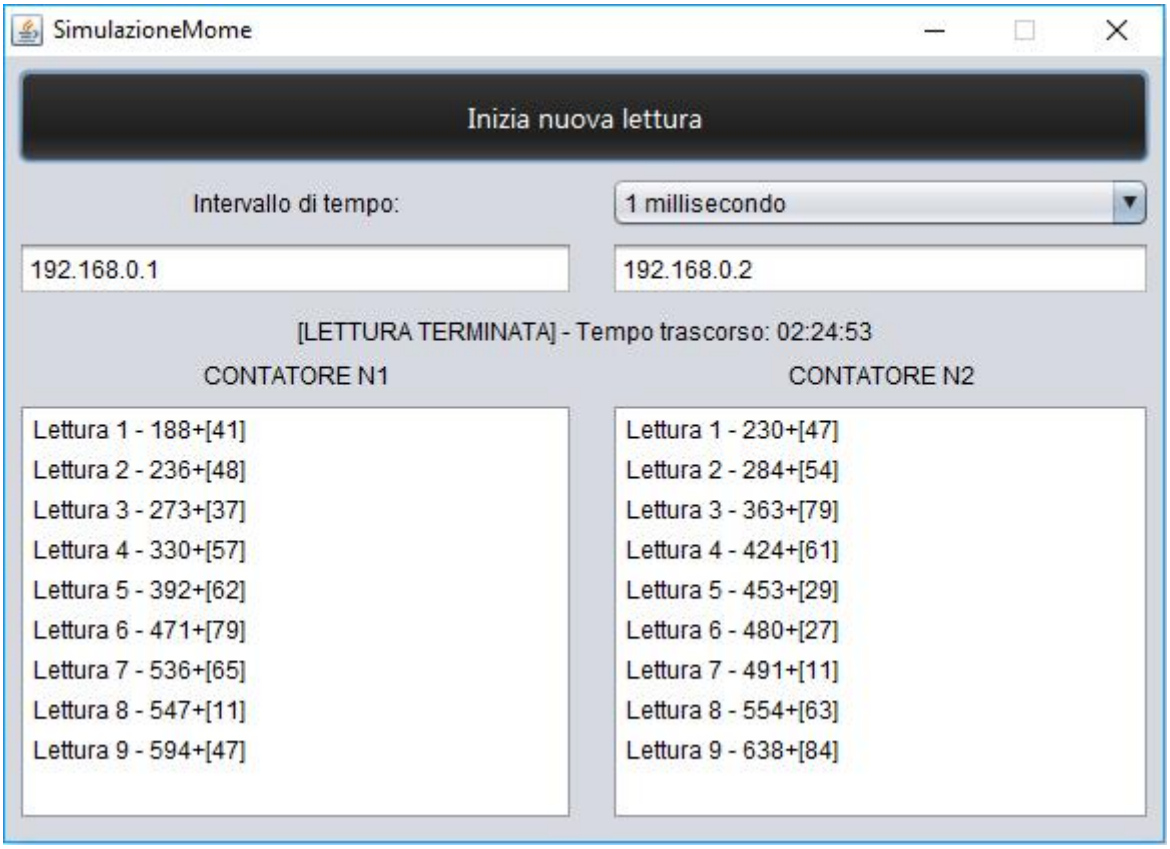
Prima di premere "Inizia la lettura" è necessario stabilire la durata del secondo, come impostazioni predefinita è impostato il secondo effettivo (1000ms), comunque può essere scelto 100ms, 10ms e 1ms.
Questa funzione è stata aggiunta per facilitare la verifica dell'esercitazione, poiché 15 minuti non sono pochi.
Sotto l'*Intervallo di tempo* si trovano 2 caselle di testo (o, come java le chiama, "FieldText"); con queste è possibile impostare un ip address differente da quello specificato nel database di esempio.

Quindi, se si inserisse nel database un nuovo contatore con IP "100.0.0.2" allora si dovrebbe modificare il primo IP con quest'ultimo, se si volessero immettere le registrazioni in quel determinato contatore.
Sotto queste caselle di testo è presente il tempo trascorso, che si incrementerà di "secondo" in "secondo" (in base all'impostazioni "Intervallo di tempo" impostato).

In basso sono presenti 2 listbox, la prima listbox è per il primo contatore mentre la seconda listbox è per il secondo contatore; quindi vengono elencate di volta in volta le varie letture effettuate.
Ogni lettura ha il *numero della lettura*, il *totale del contatore* (consumo assoluto) ed il *valore aggiuntivo*, quindi il consumo relativamente a quella lettura (consumo relativo).

Una volta premuto il pulsante "Termina lettura" (quindi dopo averla precedentemente

iniziata) il tempo si fermerà, mostrando nella label del tempo trascorso il messaggio aggiuntivo di "[LETTURA TERMINATA]" e il pulsante di inizio lettura viene impostato a "Inizia nuova lettura".



Sono state utilizzate alcune classi, quali Timer e TimerTask per poter ripetere l'operazione ogni 15minuti (se "Intervallo di tempo" è impostato ad 1 secondo).

Inoltre per effettuare la richiesta è stato inviato il carattere "?" (identifica una richiesta) preceduto dal consumo relativo ed è stato ricevuto, in risposta, il consumo assoluto.

Manuale utente

Come avviare il programma:

Molto semplicemente è necessario scegliere l'intervallo di tempo, impostato in maniere predefinita come "1 secondo" e poi è necessario impostare gli IP address (dopo aver importato o creato il database).
Infine è sufficiente premere su "Inizia lettura".

Come terminare una lettura:

Una volta premuto su "Inizia lettura" questo pulsante diventerà "Termina lettura", quindi è sufficiente premere quel suddetto pulsante.

Come e dove leggere il consumo dei contatori:

Sono presenti due listbox, la prima si riferisce al primo contatore mentre la seconda listbox si riferisce al secondo contatore. Quindi per leggere il consumo del primo contatore vedere la prima listbox, si vedrà il **numero della lettura**, il **consumo totale** ed il **consumo relativamente a quella lettura**; un esempio potrebbe essere: **Lettura 1 - 20+[20]** e **Lettura 2 - 43+[23]** il che significa che nella Lettura 2 è stato consumato un totale di 43 e

23 è il consumo relativamente a quella lettura (quindi il consumo prima della lettura sarebbe 43-23=20, esattamente il valore della Lettura 1).

Codice

Window:

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package mome;

import java.io.IOException;
import java.net.SocketException;
import java.net.SocketTimeoutException;
import java.net.UnknownHostException;
import java.util.Random;
import java.util.Scanner;
import java.util.Timer;
import java.util.TimerTask;
import javax.swing.DefaultListModel;

/**
 *
 * @author corso
 */
public class Window extends javax.swing.JFrame {

    /**
     * Creates new form Window
     */
    public Window() {
        initComponents();
    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        jButton1 = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jList1 = new javax.swing.JList<>();
        jScrollPane2 = new javax.swing.JScrollPane();
        jList2 = new javax.swing.JList<>();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jComboBox1 = new javax.swing.JComboBox();
        jTextField1 = new javax.swing.JTextField();
        jTextField2 = new javax.swing.JTextField();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("SimulazioneMome");
        setMaximumSize(new java.awt.Dimension(584, 329));
        setMinimumSize(new java.awt.Dimension(584, 329));
        setResizable(false);

        jButton1.setBackground(new java.awt.Color(0, 0, 0));
        jButton1.setFont(new java.awt.Font("Ebrima", 0, 14)); // NOI18N
```

```
        jButton1.setForeground(new java.awt.Color(255, 255, 255));
        jButton1.setText("Inizia lettura");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText("Tempo trascorso: 00:00:00");

        jScrollPane1.setViewportView(jList1);

        jScrollPane2.setViewportView(jList2);

        jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel2.setText("CONTATORE N1");

        jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel3.setText("CONTATORE N2");

        jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel4.setText("Intervallo di tempo:");

        jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "1
millisecondo", "10 millisecondi", "100 millisecondi", "1 secondo (Default)" }));
        jComboBox1.setSelectedIndex(3);

        jTextField1.setText("192.168.0.1");
        jTextField1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jTextField1ActionPerformed(evt);
            }
        });

        jTextField2.setText("192.168.0.2");

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING)
                    .addComponent(jLabel1,
javax.swing.GroupLayout.Alignment.TRAILING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(jButton1, javax.swing.GroupLayout.DEFAULT_SIZE,
564, Short.MAX_VALUE)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayo
ut.Alignment.TRAILING)
                            .addComponent(jTextField1,
javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel4,
javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jLabel2,
javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jScrollPane1))
                        .addGap(18, 18, 18)
                        .addGroup(layout.createParallelGroup(javax.swing.GroupLayo
```

```
ut.Alignment.LEADING, false)
                            .addComponent(jScrollPane2)
                            .addComponent(jLabel3,
javax.swing.GroupLayout.DEFAULT_SIZE, 270, Short.MAX_VALUE)
                            .addComponent(jComboBox1, 0,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(jTextField2))))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE,
48, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.BASELINE)
                    .addComponent(jLabel4)
                    .addComponent(jComboBox1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.BASELINE)
                    .addComponent(jTextField1,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jTextField2,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D, 9, Short.MAX_VALUE)
                .addComponent(jLabel1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.BASELINE)
                    .addComponent(jLabel2)
                    .addComponent(jLabel3))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATE
D)
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Align
ment.LEADING, false)
                    .addComponent(jScrollPane1,
javax.swing.GroupLayout.DEFAULT_SIZE, 209, Short.MAX_VALUE)
                    .addComponent(jScrollPane2))
                .addGap(10, 10, 10))
        );

        pack();
    }// </editor-fold>

    private DefaultListModel lista1=new DefaultListModel();
    private Integer lettura1=0;
    private DefaultListModel lista2=new DefaultListModel();
    private Integer lettura2=0;
    static int interval;
    static Timer timer;
    Server echoserver;

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        if(jButton1.getText()=="Inizia lettura" || jButton1.getText()=="Inizia
nuova lettura")
```

```
        {
            //se il bottone è uguale a "Inzia lettura" o "Inizia nuova lettura"
            //quindi inizierà il timer
            try
            {
                //lancio il srver
                echoserver = new Server(7);
                echoserver.start();
            }
            catch (IOException exception)
            {
                System.err.println("!! Errore: Il server non può essere avviato
automaticamente. !!\n!! Avviarlo manualmente. !!");
            }

            Integer intervalloTempo=1000;//imposto l'intervallo di tempo
predefinito a 1 secondo
            //imposto l'intervallo di tempo in base alla scelta della combobox
            if(jComboBox1.getSelectedItem()=="1 secondo (Default)")
intervalloTempo=1000;
            else if(jComboBox1.getSelectedItem()=="1 millisecondo")
intervalloTempo=1;
            else if(jComboBox1.getSelectedItem()=="10 millisecondi")
intervalloTempo=10;
            else if(jComboBox1.getSelectedItem()=="100 millisecondi")
intervalloTempo=100;
            else if(jComboBox1.getSelectedItem()=="1000 millisecondi")
intervalloTempo=1000;

            //imposto il bottone come "Termina lettura"
            jButton1.setText("Termina lettura");
            jComboBox1.setEnabled(false);//disabilita il bottone

            jTextField1.setEnabled(false);//disabilito le textfield
            jTextField2.setEnabled(false);

            Scanner sc = new Scanner(System.in);
            Integer delay = 0;//tempo di DELAY iniziale
            Integer period = intervalloTempo;//intervallo in millisecondi
            timer = new Timer();
            interval = 1;
            lettura1=0;
            lettura2=0;
            lista1.clear();
            jList1.setModel(lista1);
            lista2.clear();
            jList2.setModel(lista2);
            timer.scheduleAtFixedRate(new TimerTask()
            {
                Integer tempoTrascorso=0;
                Integer ore=0;
                Integer min=0;
                Integer sec=0;
                public void run()
                {
                    //tutto il ciò che deve ripetere

                    tempoTrascorso++;
                    //System.out.println(tempoTrascorso);
                    String contatore="00:00:00";
                    sec++;
                    if(sec>59)
                    {
                        sec=0;
                        min++;
                        if(min>59)
```

```
                        {
                            min=0;
                            ore++;
                        }
                    }
                    //stampo l'ora, i minuti e i secondi nella label
                    String h="", m="", s="";
                    if(ore<10) h="0"+ore;
                    else h=""+ore;
                    if(min<10) m="0"+min;
                    else m=""+min;
                    if(sec<10) s="0"+sec;
                    else s=""+sec;
                    contatore="Tempo trascorso: "+h+":"+m+":"+s;
                    jLabel1.setText(contatore);
                    //900 --> 15min*60sec = 900secondi === se il tempo trascorso
diviso 15min è uguale a 0, vuol dire che è un multiplo di 15 minuti -> quindi
lettura
                    if(tempoTrascorso%900==0)
                    {
                        //genero i numeri random
                        Integer n1, n2;
                        Random random=new Random();
                        Integer consumoMAX=100;
                        n1=random.nextInt(consumoMAX);
                        n2=random.nextInt(consumoMAX);
                        letture(n1, n2);
                    }
                }
            }, delay, period);
        }
        else
        {
            //quando il bottone risulta essere uguale a "Termina lettura"
            jButton1.setText("Inizia nuova lettura");
            timer.cancel();
            jLabel1.setText("[LETTURA TERMINATA] - "+jLabel1.getText());
            jComboBox1.setEnabled(true);
            jTextField1.setEnabled(true);
            jTextField2.setEnabled(true);
            try
            {
                echoserver.interrupt();
                echoserver.join();
            }
            catch (InterruptedException exception)
            {
                //System.err.println("Errore!");
            }
        }
    }

    private void jTextField1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    public void letture(Integer num1, Integer num2)
    {
        //ottengo gli ip dalle textfield
        String ipaddress1=jTextField1.getText();
        String ipaddress2=jTextField2.getText();
        String address="127.0.0.1";//ip address del server
        int porta;
        String request, answer;
        Client client, client2;
        porta = 7;
```

```
        //IP_ADDRESS(spazio)CONSUMO_DA_AGGIUNGERE
        try
        {
            request=ipaddress1+" "+num1+"   ?";
            client = new Client();
            //credo la risposta inviando la richiesta al server
            answer = client.sendAndReceive(request, address, porta);
            //System.out.println("GUI - Ricevuto in risposta: \"" +answer+"\"");

            lettura1++;//incremento la lettura1 e controllo se abbia ricevuto un
errore come risposta (E)
            if(!answer.equals("E")) lista1.addElement("Lettura "+lettura1+" -
"+answer+"+["+num1+"]");
            else lista1.addElement("Lettura "+lettura1+" - ERRORE");
            jList1.setModel(lista1);
            client.close_socket();

            client2 = new Client();
            request=ipaddress2+" "+num2+"                        ";
            answer = client2.sendAndReceive(request, address, porta);
            //System.out.println("GUI - Ricevuto in risposta: \"" +answer+"\"");
            lettura2++;//incremento la lettura 2
            if(!answer.equals("E")) lista2.addElement("Lettura "+lettura2+" -
"+answer+"+["+num2+"]");
            else lista2.addElement("Lettura "+lettura2+" - ERRORE");
            jList2.setModel(lista2);
            client2.close_socket();
        }
        catch (SocketException exception)
        {
            System.err.println("!! Errore: Creazione socket !!");
        }
        catch (UnknownHostException exception)
        {
            System.err.println("!! Indirizzo IP errato !!");
        }
        catch (SocketTimeoutException exception)
        {
            System.err.println("!! Nessuna risposta dal server !!");
        }
        catch (IOException exception)
        {
            System.err.println("!! Errore: Generico di comunicazione !!");
        }
    }
    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the
default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
```

```
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Window.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Window.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Window.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Window.class.getName()).log(java.util.logging.L
evel.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Window().setVisible(true);
            }
        });

        //impostare le letture precedenti (?) importandole dal database: bisogna
anche settare il numero delle letture già presenti
    }

    // Variables declaration - do not modify
    private javax.swing.JButton jButton1;
    private javax.swing.JComboBox jComboBox1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JList<String> jList1;
    private javax.swing.JList<String> jList2;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    // End of variables declaration
}
```

Client

```
package mome;

import java.net.*;
import java.io.*;

public class Client
{
    private DatagramSocket socket;

    public Client() throws SocketException
    {
        socket = new DatagramSocket();
        socket.setSoTimeout(1000); // 1000ms = 1s
    }

    public void close_socket()
    {
        socket.close();
```

```
    }

    public String sendAndReceive(String request, String host, int port) throws
UnknownHostException, IOException, SocketTimeoutException
    {
        byte[] buffer;
        DatagramPacket datagram;
        String answer;
        // indirizzo IP del destinatario del datagram
        InetAddress address = InetAddress.getByName(host);

        // verifica chiusura socket
        if(socket.isClosed())
        {
            throw new IOException();
        }
        // trasformazione in array di byte della stringa
        buffer = request.getBytes("UTF-8");
        // costruzione datagram di richiesta
        datagram = new DatagramPacket(buffer, buffer.length, address, port);
        socket.send(datagram); // trasmissione datagram di richiesta
        // attesa ricezione datagram di richiesta (tempo massimo di attesa: 1s)
        socket.receive(datagram);
        // verifica indirizzo/porta provenienza datagram di risposta
        if(datagram.getAddress().equals(address) && datagram.getPort() == port)
        {
            answer = new String( datagram.getData(), 0, datagram.getLength(),
"ISO-8859-1");
        }
        else
        {
            throw new SocketTimeoutException();
        }
        return answer;
    }
}
```

Server

```
package mome;

import java.net.*;
import java.io.*;
import java.util.*;
import com.mysql.jdbc.Connection;
import com.mysql.jdbc.Statement;
import com.mysql.jdbc.exceptions.MySQLSyntaxErrorException;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.text.SimpleDateFormat;

public class Server extends Thread
{
    private DatagramSocket socket;

    public Server(int port) throws SocketException
    {
        socket = new DatagramSocket(port);
        socket.setSoTimeout(1000); // 1000ms = 1sec
    }

    public void run()
    {
        DatagramPacket answer;
        // creazione di un array di byte della dimensione specificata
        byte[] buffer = new byte[24];//8bit(1Byte)*3=3Byte
```

```
        // creazione di un datagram UDP a partire dall'array di byte
        DatagramPacket request = new DatagramPacket(buffer, buffer.length);

        while (!Thread.interrupted())
        {
            try
            {
                // attesa ricezione datagram di richiesta (tempo massimo di
attesa: 1s)
                socket.receive(request);
                // costruzione datagram di risposta (identico al datagram di
richiesta)
                answer = new DatagramPacket( request.getData(),
request.getLength(), request.getAddress(), request.getPort());

                String[] array =new String(answer.getData(), answer.getOffset(),
answer.getLength()).split(" ");
                String address=array[0];
                Integer consumoAdd=Integer.parseInt(array[1]);
                Integer consumo=0;
                String data=new SimpleDateFormat("yyyy-MM-dd").format(new
Date());//specifico il tipo di data (formato)
                String ora=new SimpleDateFormat("HH:mm:ss").format(new
Date());//specifico il formato del tempo

                Connection con = null;
                Statement st=null;
                ResultSet rs=null;
                Boolean errore=false;

                String url="jdbc:mysql://localhost:3306/";
                String user="root", db="mome", password="";
                try
                {

con=(Connection)DriverManager.getConnection(url+db,user,password);
                    //System.out.println("Connessione riuscita.");
                    String sql="SELECT ultima_rilevazione FROM contatori WHERE
ip_address='"+address+"'";
                    st=(Statement)con.createStatement();
                    rs=st.executeQuery(sql);
                    while(rs.next())
                        consumo=rs.getInt("ultima_rilevazione");
                }catch(SQLException e)
                {
                    System.err.println("!! Errore: "+e+" !!");
                    errore=true;
                }
                consumo+=consumoAdd;
                try
                {

con=(Connection)DriverManager.getConnection(url+db,user,password);
                    //System.out.println("Connessione riuscita.");
                    String sql="UPDATE contatori SET
ultima_rilevazione='"+consumo+"' WHERE ip_address='"+address+"'";
                    st=(Statement)con.createStatement();
                    st.executeUpdate(sql);
                    sql="INSERT INTO registrazioni(id, contatore_ip, data,
consumo) VALUES(NULL, '"+address+"', '"+data+" "+ora+"', '"+consumo+"')";
                    st=(Statement)con.createStatement();
                    st.executeUpdate(sql);
                }catch(SQLException e)
                {
                    System.err.println("!! Errore: "+e+" !!");
                    errore=true;
```

```
                }

                String stringa="E";//se si è verificato un errore
                if(!errore) stringa=consumo+"";
                byte[] buffer_risposta = new byte[24];//8bit(1Byte)*3=3Byte
                buffer_risposta = stringa.getBytes("UTF-8");
                DatagramPacket risposta = new DatagramPacket(buffer_risposta,
buffer_risposta.length);
                // trasmissione datagram di risposta
                answer = new DatagramPacket(risposta.getData(),
risposta.getLength(), request.getAddress(), request.getPort());
                socket.send(answer);
            }
            catch (IOException exception)
            {
            }
        }
        socket.close(); // chiusura del socket
    }
}
```