```java
/*
Δίκτυα Ι
Αμαραντίδου Ευθυμία
ΑΕΜ: 9762
 */

package com.networks;

import ithakimodem.*;

import javax.imageio.ImageIO;
import javax.swing.plaf.basic.BasicButtonUI;
import java.awt.*;
import java.util.*;
import java.util.List;
import java.util.concurrent.TimeUnit;
import java.io.*;
import java.awt.image.BufferedImage;
import java.nio.charset.StandardCharsets;

/*~
Class userApplication implements an application which connects
to server "ithaki" through a virtual modem and takes statistical
measurements of specific connection variables (response time, BER)
 */
public class userApplication {
    public static void main(String[] args) throws IOException {
        Modem modem = new Modem(8000);
        modem.setTimeout(2000);
        modem.open("ithaki");

        (new userApplication()).initialization(modem, "ATD2310ITHAKI\r");
        (new userApplication()).echo(modem, " E7536\r");
        (new userApplication()).get_image(modem, "G2973" + "CAM=PTZ\r", "image_with_error_v2-Session1");
        (new userApplication()).get_gps_coordinates(modem, "P6519\r");
        (new userApplication()).get_gps_image(modem, "P6519", "gps_image-Session1");
        (new userApplication()).arq(modem, "Q9405\r", "R8482\r");

        modem.close();
    }

    /*
     initialization: Establishes the connection with the virtual modem.
    */
    public void initialization(Modem modem, String address) {

        int sym;
        modem.write(address.getBytes());

        for(;;) {
            try{
                sym = modem.read();
                if(sym == -1) break;

                System.out.print(((char) sym));

            } catch (Exception x) {
                break;
            }
        }
    }

    /*!
     echo: Requests echo packets from the server and calculates the response time.
    */
    public void echo(Modem modem, String address) throws IOException {
        int sym;
```

```
    int packet_counter = 0;
    double start;
    double tic = 0;
    double tac = 0;
    int duration;
    String message = "";
    List<Integer> response_times = new ArrayList<>();

    start = System.currentTimeMillis();

    while((int)(tac - start) < 240000){
        modem.write(address.getBytes());
        packet_counter ++;
        tic = System.currentTimeMillis();
        for(;;) {
            try {
                sym = modem.read();
                message += (char) sym;

                if (sym == -1) break;

                if (message.equals("PSTART ")) {
                    tic = System.currentTimeMillis();
                    message = "";
                }

                if (message.contains(" PSTOP")) {
                    System.out.println(message);
                    tac = System.currentTimeMillis();
                    duration = (int) (tac - tic);
                    response_times.add(duration);
                    System.out.println("Response time = " + duration + "ms");
                    message = "";
                }
            } catch (Exception e) {
                break;
            }
        }
    }

    System.out.println("Response times = " + response_times + "\nNum of packets = " + packet_counter);

    FileWriter writer = null;
    try {
        writer = new FileWriter("echo-response-Session1.csv");
    } catch (IOException e) {
        e.printStackTrace();
    }
    for (Integer response_time : response_times) {
        assert writer != null;
        writer.write(response_time.toString() + System.lineSeparator());
    }
    assert writer != null;
    writer.close();
}

/*!
 get_image: Requests images from the server (error free or not depending on the variable: address).
 */
public void get_image(Modem modem, String address, String file_name) throws IOException{
    modem.write(address.getBytes());
    byte[] imageBytes = new byte[80000];
    byte b;
    int i = 1;
    boolean image_started = false;
    imageBytes[0] = (byte)modem.read();
```

```
    for(;;){
        try{
            b = (byte)modem.read();
            System.out.println(b);
            imageBytes[i++] = b;

            if(imageBytes[i-2] == (byte)0xff & imageBytes[i-1] == (byte)0xD8 & !image_started) {
                image_started = true;
            }

            if(imageBytes[i-2] == (byte)0xff & imageBytes[i-1] == (byte)0xD9){
                break;
            }

        }catch(Exception e){
            break;
        }
    }

    ByteArrayInputStream image = new ByteArrayInputStream(imageBytes);
    BufferedImage myImage = ImageIO.read(image);
    ImageIO.write(myImage, "jpeg", new File(file_name + ".jpeg"));
}

/*!
 get_gps_coordinates: Calculates multiple longtitude and latitude values of traces.
 */
public String get_gps_coordinates(Modem modem, String address) throws IOException {

    String gps_request_code = address + "R=1010050" + "\r";
    System.out.println(gps_request_code);
    modem.write(gps_request_code.getBytes());

    int b;
    String info = "";
    String request_image = "";
    String longtitude = "";
    int longtitude_sec;
    String latitude = "";
    int latitude_sec;
    String substring = "";

    ArrayList<String> message = new ArrayList<>();

    for(;;){
        try{
            b = modem.read();

            info += (char)b;

            if(info.equals("START ITHAKI GPS TRACKING\r\n")){
                info = "";
            }

            if(info.contains("STOP ITHAKI GPS TRACKING\r\n")){
                break;
            }

            if(info.contains("\r\n")){
                System.out.println("" + info);
                if(info.contains("GPGGA")) {
                    request_image = request_image + "T=";
                    longtitude = info.substring(31, 35);
                    longtitude_sec = Integer.parseInt(info.substring(36, 40));
                    longtitude_sec *= 0.006;
                    latitude = info.substring(18, 22);
                    latitude_sec =  Integer.parseInt(info.substring(23, 27));
```

```
                latitude_sec *= 0.006;
                request_image += longtitude + longtitude_sec + latitude + latitude_sec;
                message.add(info);
            }
            info = "";
        }

    }catch(Exception e){
        break;
    }
}

System.out.println("" + request_image);
System.out.println("Message = \n" + message);
return request_image;
}

/*!
 get_gps_image: Requests Google Map image with the traces.
 */
public void get_gps_image(Modem modem, String address, String file_name) throws IOException {

String coordinates = (new userApplication()).get_gps_coordinates(modem, address);

String request_image = address + coordinates + "\r";
System.out.println(request_image);

int bytes_num = 0;
String outputFile = file_name + ".jpg";

modem.write(request_image.getBytes());
try(OutputStream outputStream = new FileOutputStream(outputFile);)
{
    int b;
    while ((b = modem.read()) != -1){
        //System.out.println("ByteRead = " + byteRead);
        outputStream.write(b);
        bytes_num++;
    }
    System.out.println("Number of bytes: " + bytes_num);
}
catch(Exception ex) {
    ex.printStackTrace();
}
}

/*!
arq: Implements ARQ mechanisms. Receives packages, checks if errors occurred and requests retransmission if needed.
 */
public void arq(Modem modem, String address_ack, String address_nack) throws IOException {

int sym;
double start;
double tic = 0;
double tac = 0;
int duration;
List<Integer> response_times = new ArrayList<>();
List<Integer> repeats = new ArrayList<>();
boolean error_free = false;
int nack = 0;
int xor = 0;

String packet = "";
String info = "";
int FCS = 0;

start = System.currentTimeMillis();
```

```
while((int)(tac - start) < 246000) {
    System.out.println("hi!");
    xor = 0;

    if(nack > 0){
        modem.write(address_nack.getBytes());
    }
    else{
        modem.write(address_ack.getBytes());
    }

    for (; ; ) {
        try {
            sym = modem.read();
            packet += (char)sym;

            if (sym == -1) {
                System.out.println(sym);
                break;
            }

            if (packet.contains("PSTART ")) {
                tic = System.currentTimeMillis();
                packet = "";
            }

            if (packet.contains("PSTOP")) {
                break;
            }

        } catch (Exception e) {
            System.out.println("OOPS! Something went wrong!");
            break;
        }
    }

    System.out.println(packet);

    info = packet.substring(packet.indexOf("<") + 1, packet.indexOf(">"));

    System.out.println(info);

    FCS = Integer.parseInt(packet.substring(packet.indexOf(">") + 2, packet.indexOf(">") + 5));

    packet = "";

    for (char c : (info.toCharArray())) {
        xor = xor ^ (int) c;
    }

    if (xor == FCS) {
        System.out.println("Hi");
        tac = System.currentTimeMillis();
        duration = (int) (tac - tic);
        response_times.add(duration);

        error_free = true;
        repeats.add(nack);
        nack = 0;
    } else {
        nack++;
        error_free = false;
    }
}

FileWriter writer = null;
```

```
        try {
            writer = new FileWriter("Response_times.csv");
        } catch (IOException e) {
            e.printStackTrace();
        }

        for (Integer response_time : response_times) {
            assert writer != null;
            writer.write(response_times.toString() + System.lineSeparator());
        }
        writer.close();

        try {
            writer = new FileWriter("Repeats.csv");
        } catch (IOException e) {
            e.printStackTrace();
        }

        for (Integer repeat : repeats) {
            assert writer != null;
            writer.write(repeats.toString() + System.lineSeparator());
        }
        writer.close();
    }
}
```