```java
package src.com.networks;

import ithakimodem.*;
import java.io.*;
import java.util.ArrayList;
import java.util.Arrays;

public class userApplication {

    public static void main(String[] args) throws IOException{
        Modem modem = new Modem(12000);
        modem.setTimeout(8000);
        modem.open("ithaki");

        int numOfT = 60; // number of points for gps

        String init_code = "ATD2310ITHAKI\r";
        String echo_code = "E3484\r";
        String video_code = "M6027 CAM=PTZ\r"; //CAM=PTZ DIR=D
        String video_errors_code = "G2436\r";
        String gps_code = "P9302";
        String gps_code_T = "P9302"+"R=10120"+String.valueOf(numOfT)+"\r";
        String ack_code = "Q5860\r";
        String nack_code = "R7086\r";

        (new userApplication()).initialization(modem, init_code);
        //(new userApplication()).echo(modem, echo_code);
        //(new userApplication()).video(modem, video_code);
        //(new userApplication()).video(modem, video_errors_code);
        //(new userApplication()).gps_image(modem, (new userApplication()).gps_T((new userApplication()).gps(modem,
gps_code_T)), gps_code);
        (new userApplication()).arq_result(modem, ack_code, nack_code);

        modem.close();
    }

    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;
            }
        }
    }

    public void echo(Modem modem, String address) throws IOException{
        int sym;
        double tic;
        double tac;
        int duration;
        long start;
        int[] delays = {};

        String message = "";

        start = System.currentTimeMillis();
```

```
        tic = 0;
        tac = 0;

        while((int)(tac - start) < 300000){
            modem.write(address.getBytes());
            for(;;){
                try{
                    sym = modem.read();
                    if(sym == -1){
                        break;
                    }
                    message += (char)sym;

                    if(message.contains("PSTART ")){
                        message = "";
                        tic = System.currentTimeMillis();
                        System.out.println("Tic = " + tic);
                    }

                    if(message.contains(" PSTOP")){
                        System.out.println(message);
                        message = "";
                        tac = System.currentTimeMillis();
                        break;
                    }
                }
                catch(Exception ex) {
                    break;
                }
            }
            duration = (int)(tac - tic);
            delays = Arrays.copyOf(delays, delays.length + 1);
            delays[delays.length - 1] = duration;
            System.out.println("Response time = " + duration + "ms");
        }
        try (FileWriter pr = new FileWriter("Response_times_echo.csv")){
            for (int j = 0; j < delays.length; j++){
                pr.append(String.valueOf(j+1)+","+String.valueOf(delays[j]));
                pr.append("\n");
            }
            pr.close();
        }
    }

    public void video(Modem modem, String address) throws IOException{
        // byte[] buffer = new byte[BUFFER_SIZE];
        int count_bytes = 0;
        String outputFile = "Frame.jpg";

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

    public StringBuilder gps(Modem modem, String address) throws IOException{
        modem.write(address.getBytes());
```

```
    int gps;
    StringBuilder message = new StringBuilder();
    for(;;)
    {
        try{
            gps = modem.read();
            if(gps == -1){
                break;
            }
            message.append((char)gps); //append((char)gps);
            //System.out.println(message);

            if(message.toString().contains("START ITHAKI GPS TRACKING\r\n")){
                message.delete(0, message.length());
            }

            if(message.toString().contains("STOP ITHAKI GPS TRACKING\r\n")){
                message.delete(message.length()-28, message.length());
                break;
            }
        }
        catch(Exception ex) {
            break;
        }
    }
    System.out.println(message);
    return message;
}

public String[] gps_T(StringBuilder message) throws IOException{

    String T_messages[] = {};
    String messageN = "N";
    String messageE = "E";
    double sec_const = 0.006;

    System.out.println(message.length());
    for (int i = 0; i < message.length(); i++)
    {
        if (message.charAt(i) == 'N' && message.charAt(i-1) == ',')
        {
            messageN = message.toString().substring(i-10, i-6);
            int sec = Integer.parseInt(message.toString().substring(i-5, i-1));
            System.out.println(sec);
            sec *= sec_const;
            messageN += String.valueOf(sec);
        }
        if (message.charAt(i) == 'E' && message.charAt(i-1) == ',')
        {
            messageE = message.toString().substring(i-10, i-6);
            int sec = Integer.parseInt(message.toString().substring(i-5, i-1));
            System.out.println(sec);
            sec *= sec_const;
            messageE += String.valueOf(sec);

            T_messages = Arrays.copyOf(T_messages, T_messages.length + 1);
            T_messages[T_messages.length - 1] = messageE+messageN;

            System.out.println(T_messages[T_messages.length - 1]);
        }
    }
    return T_messages;
}

public void gps_image(Modem modem, String[] T_messages,  String address)
{
```

```
    int count_bytes = 0;
    String outputFile = "GPS.jpg";

    for (int i = 0; i < 54; i+= 6)
    {
        address += "T=";
        address += T_messages[i];
    }
    System.out.println(address);
    for (int i = 0; i < address.length(); i++)
    {
        if (address.charAt(i) == '.')
        {
            address = address.replace(".", "");
        }
    }
    address += "\r";
    System.out.println(address);
    modem.write(address.getBytes());
    try(OutputStream outputStream = new FileOutputStream(outputFile);)
    {
        int byteRead;
        while ((byteRead = modem.read()) != -1){
            //System.out.println("ByteRead = " + byteRead);
            outputStream.write(byteRead);
            count_bytes++;
        }
        System.out.println("Number of bytes: " + count_bytes);
    }
    catch(Exception ex) {
        ex.printStackTrace();
    }
}

public void arq_result(Modem modem, String ack_code, String nack_code) throws IOException
{
    int sym = -1;
    double tic;
    double tac;
    int duration;
    long start;
    int[] responses = {};
    boolean correct = false;
    int repeat = 0;
    ArrayList<String> content = new ArrayList<String>();
    ArrayList<Integer> FCS = new ArrayList<Integer>();
    ArrayList<Integer> repeat_until_correct = new ArrayList<Integer>();

    String message = "";

    start = System.currentTimeMillis();

    tic = 0;
    tac = 0;

    while((int)(tac - start) < 300000){

        int xor_result = 0;

        if (correct == true)
        {
            modem.write(ack_code.getBytes());
        }
        else
        {
            modem.write(nack_code.getBytes());
            repeat++;
```

```
        }
        for(;;){
            try{
                sym = modem.read();
                if(sym == -1){
                    break;
                }
                message += (char)sym;

                if(message.contains("PSTART ")){
                    message = "";
                    tic = System.currentTimeMillis();
                    // System.out.println("Tic = " + tic);
                }

                if(message.contains(" PSTOP")){
                    break;
                }
            }
            catch(Exception ex) {
                break;
            }
        }
        if (sym == -1)
        {
            return;
        }
        System.out.println("\n" + message);
        String content_temp = message.substring(message.indexOf("<")+1, message.indexOf(">"));
        System.out.println(content_temp);
        content.add(content_temp);

        int FCS_temp = Integer.parseInt(message.substring(message.indexOf(">")+2, message.indexOf(">")+5));
        System.out.println(FCS_temp);
        FCS.add(FCS_temp);

        message = "";

        for (char c : (content.get(content.size() - 1)).toCharArray())
        {
            xor_result = xor_result^(int) c;
        }
        // System.out.println("This is an XOR result: " + xor_result + " and this is the FCS result: " + FCS.get(FCS.size() - 1));

        if (xor_result == FCS.get(FCS.size() - 1))
        {
            tac = System.currentTimeMillis();
            duration = (int)(tac - tic);
            responses = Arrays.copyOf(responses, responses.length + 1);
            responses[responses.length - 1] = duration;
            System.out.println("Response time = " + duration + "ms");
            correct = true;

            repeat_until_correct.add(repeat);
            repeat = 0;
        }
        else
        {
            correct = false;
            repeat++;
        }
    }
    System.out.println("\n" + content);
    System.out.println("\n" + FCS);
    System.out.println("\n" + repeat_until_correct);
    System.out.println("\n" + "Length of responses array: " + responses.length);
```

```
        try (FileWriter pr = new FileWriter("Response_times_arq.csv"))
        {
            for (int j = 0; j < responses.length; j++){
                pr.append(String.valueOf(j+1)+","+String.valueOf(responses[j]));
                pr.append("\n");
            }
            pr.close();
        }
        try (FileWriter pr = new FileWriter("Repeats_arq.csv"))
        {
            for (int j = 0; j < repeat_until_correct.size(); j++){
                pr.append(String.valueOf(j+1)+","+String.valueOf(repeat_until_correct.get(j)));
                pr.append("\n");
            }
            pr.close();
        }
    }
}
```