```java
/* Δικτυα Υπολογιστών Ι
 * Εργασία 2016-17
 * Μπαλτζής Ευριπίδης
 * AEM: 8196
 * mail: eurobaltzis@gmail.com
 */


import java.io.*;
import java.lang.System;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import ithakimodem.*;


public class userApplication{

    String outMessage=null;
    String inMessage=null;
    String finalMessage="\r\n\n\n";
    String pstop="PSTOP";

    public userApplication(){}

    public static void main(String[] args) {
        Modem modem=new Modem(32000);
        userApplication app=new userApplication();
        app.ithakiconnection(modem);
        app.ARQ(modem);
        app.receivePacket(modem);
        app.getImageWithoutErrors(modem);
        app.getImageWithErrors(modem);
        app.getGPSTrace(modem);
    }


    public void ithakiconnection(Modem modem) {
        outMessage="ATD2310ITHAKI\r";
        modem.write(outMessage.getBytes());
        for(;;){
            int input=modem.read();
            inMessage+=(char)input;
            System.out.print((char)input);
            if(inMessage.contains(finalMessage)){
                break;
            }
```

```
    }
}

public void receivePacket(Modem modem){

    long packetStartTime;
    long response;
    long currentTime;
    File graph1 = new File("C:\\Users\\Evripidis\\Desktop\\graph11.txt");

    try {
        FileWriter fileWriter = new FileWriter(graph1);
        long startTime=System.currentTimeMillis();
        while(System.currentTimeMillis()<=startTime+240000) {
            outMessage="E0790\r";
            modem.write(outMessage.getBytes());
            inMessage="";
            packetStartTime=System.currentTimeMillis();
            for(;;) {
                int input=modem.read();
                inMessage+=(char)input;
                System.out.print((char)input);
                if(inMessage.contains(pstop)){
                    break;
                }
            }
            System.out.print(" Package received.\n");
            currentTime=System.currentTimeMillis();
            response=currentTime-packetStartTime;
            String content = String.valueOf(response);
            fileWriter.write(content+"\r\n");
        }
        fileWriter.close();
    }
    catch (IOException io) {
        io.printStackTrace();
    }
}

public void getImageWithoutErrors(Modem modem){
    File imageWithoutErrors = new File("C:\\Users\\Evripidis\\Desktop\\imageWithoutErrors1.jpeg");
    try {
        FileOutputStream fileWriter = new FileOutputStream(imageWithoutErrors);
        try {
            imageWithoutErrors.createNewFile();
            outMessage="M6746\r";
            modem.write(outMessage.getBytes());
```

```
            byte previous=0;
            byte input=(byte) modem.read();
            if (input==(byte)0xFF){
                previous = (byte)input;
                input=(byte) modem.read();
                System.out.print("Start of image delimeter 0xFF\n");
                if (input==(byte)0xD8){
                    fileWriter.write(previous);
                    fileWriter.write(input);
                    System.out.print("Start of image delimeter 0xD8\n");
                    boolean flag=false;
                    while(flag==false){
                        input=(byte) modem.read();
                        fileWriter.write(input);
                        //System.out.print("Sending image without errors\n");
                        if(input==(byte)0xD9 && previous==(byte)0xFF){
                            System.out.print("End of image delimeter 0xFF\n");
                            System.out.print("End of image delimeter 0xD9\n");
                            System.out.print("END OF IMAGE\n");
                            fileWriter.close();
                            flag=true;
                        }
                        previous=(byte)input;
                    }
                }
            }

         }
        catch  (IOException io) {
            io.printStackTrace();
        }
    }
    catch (FileNotFoundException f) {
        f.printStackTrace();
    }
    System.out.print("Image without errors sent.\n");
}

public void getImageWithErrors(Modem modem){
    File imageWithErrors = new File("C:\\Users\\Evripidis\\Desktop\\imageWithErrors1.jpeg");
    try {
        FileOutputStream fileWriter = new FileOutputStream(imageWithErrors);
        try{
            outMessage="G7681\r";
            modem.write(outMessage.getBytes());
            byte previous=(byte)modem.read();
            byte input=(byte) modem.read();
```

```
            System.out.print((byte)previous+"\n");
            int reps=0;
            while((previous!=(byte) 0xFF || input!=(byte) 0xD8) && reps<100){
                previous=(byte)input;
                System.out.print((byte)previous+"\n");
                input=(byte)modem.read();
                reps++;
                System.out.print(reps+"\n");
            }
            if(reps==100){
                System.out.print("ERROR\n");
            }
            else{
                imageWithErrors.createNewFile();
                fileWriter.write(previous);
                fileWriter.write(input);
                System.out.print("Start of image delimeter 0xFF\n");
                System.out.print("Start of image delimeter 0xD8\n");
                boolean flag=false;
                while(flag==false){
                    input=(byte) modem.read();
                    fileWriter.write(input);
                    //System.out.print("Sending image with errors\n");
                    if(input==(byte)0xD9 && previous==(byte)0xFF){
                        System.out.print("End of image delimeter 0xFF\n");
                        System.out.print("End of image delimeter 0xD9\n");
                        System.out.print("END OF IMAGE\n");
                        fileWriter.close();
                        flag=true;
                    }
                    previous=(byte)input;
                }
            }
        }
        catch  (IOException io) {
            io.printStackTrace();
        }
    }
    catch (FileNotFoundException f) {
        f.printStackTrace();
    }
    System.out.print("Image with errors sent.\n");
}

public void getGPSTrace(Modem modem)
{
```

```
boolean flag1 = false;
boolean flag2 = true;
String[] traces= new String[9];
int[] time = new int[9];
String r = "";
int numberOfTraces=0;
inMessage = "";
try {
    int x1 =(int) (Math.random()*35) +1;
    r = Integer.toString(x1);
    if(x1<10){
        r = "0" + r;
    }
    System.out.println(r);
    outMessage="P7462R=100" + r + "99\r";
    modem.write(outMessage.getBytes());
    inMessage="";
    while(numberOfTraces<9){
        int input=modem.read();
        inMessage+=(char)input;
        if(inMessage.endsWith("*") && flag1==false){

            System.out.println(inMessage);
            int startingPoint=inMessage.indexOf("$GPGGA")+ 7;

            //TIME
            int timeHours = Integer.valueOf(inMessage.substring(startingPoint, startingPoint+2));
            int timeMinutes = Integer.valueOf(inMessage.substring(startingPoint+2, startingPoint+4));
            int timeSeconds = Integer.valueOf(inMessage.substring(startingPoint+4, startingPoint+6));
            time[numberOfTraces] = timeHours*3600 + timeMinutes*60 + timeSeconds;
            System.out.println(time[numberOfTraces]);

            // LATITUDE
            startingPoint += 11;
            int latitudeDegrees = Integer.valueOf(inMessage.substring(startingPoint, startingPoint+2));
            int latitudeMinutes = Integer.valueOf(inMessage.substring(startingPoint+2, startingPoint+4));
            int latitudeSeconds = Integer.valueOf(inMessage.substring(startingPoint+5, startingPoint+9))*6/1000;
            String latitudeDirection = inMessage.substring(startingPoint+10);
            if(latitudeDirection.equalsIgnoreCase("S")){
                latitudeDegrees = (-1)*latitudeDegrees;
            }
            int latitude = latitudeDegrees*10000 + latitudeMinutes*100 +latitudeSeconds;
            String lat = Integer.toString(latitude);
```

```
// LATITUDE
startingPoint += 11;
int latitudeDegrees = Integer.valueOf(inMessage.substring(startingPoint, startingPoint+2));
int latitudeMinutes = Integer.valueOf(inMessage.substring(startingPoint+2, startingPoint+4));
int latitudeSeconds = Integer.valueOf(inMessage.substring(startingPoint+5, startingPoint+9))*6/1000;
String latitudeDirection = inMessage.substring(startingPoint+10);
if(latitudeDirection.equalsIgnoreCase("S")){
    latitudeDegrees = (-1)*latitudeDegrees;
}
int latitude = latitudeDegrees*10000 + latitudeMinutes*100 +latitudeSeconds;
String lat = Integer.toString(latitude);

// LONGTITUDE
startingPoint += 12;
int longtitudeDegrees = Integer.valueOf(inMessage.substring(startingPoint, startingPoint+3));
int longtitudeMinutes = Integer.valueOf(inMessage.substring(startingPoint+3, startingPoint+5));
int longtitudeSeconds = Integer.valueOf(inMessage.substring(startingPoint+6, startingPoint+10))*6/1000;
String longtitudeDirection = inMessage.substring(startingPoint+11);
if(longtitudeDirection.equalsIgnoreCase("W")){
    longtitudeDegrees = (-1)*latitudeDegrees;
}
int longtitude = longtitudeDegrees*10000 + longtitudeMinutes*100 +longtitudeSeconds;
String longt = Integer.toString(longtitude);

//CLEAR INMESSAGE
inMessage="";
System.out.println("Number Of Traces:" + numberOfTraces);

//CHECK
if(numberOfTraces == 0){
    traces[numberOfTraces] = longt+ lat;
    System.out.println(traces[numberOfTraces]);
    numberOfTraces++;
    flag1 = true;
}
else if(time[numberOfTraces] - time[numberOfTraces-1] >= 4){
    traces[numberOfTraces] = longt+ lat;
    System.out.println("Current Trace: " +traces[numberOfTraces]);
    for(int i=0; i<numberOfTraces; i++){
        if(traces[numberOfTraces].equals(traces[i])){
            System.out.println("is the same as trace: " + traces[i]);
            flag2 = false;
        }
    }
```

```
            if(flag2 == true){
                System.out.println(traces[numberOfTraces]);
                numberOfTraces++;
                flag1 = true;
                System.out.println("MPHKA");
            }
            flag2=true;
        }
    }
    //CHECK FOR END MESSAGE
    if(flag1==true){
        while(!(inMessage.endsWith("STOP ITHAKI GPS TRACKING\r\n"))){
            input=modem.read();
            inMessage+=(char)input;
        }
        System.out.println(inMessage);
        flag1=false;
        x1 =(int) (Math.random()*35) +1;
        r = Integer.toString(x1);
        if(x1<10){
            r = "0" + r;
        }
        System.out.println(r);
        outMessage="P7462R=100" + r + "99\r";
        modem.write(outMessage.getBytes());
        inMessage = "";
    }
    else if(flag1==false && inMessage.endsWith("STOP ITHAKI GPS TRACKING\r\n")){
        flag1=false;
        x1 =(int) (Math.random()*35) +1;
        r = Integer.toString(x1);
        if(x1<10){
            r = "0" + r;
        }
        outMessage="P7462R=100" + r + "99\r";
        System.out.println("\n\n\n" +outMessage +"\n\n\n");
        modem.write(outMessage.getBytes());
        inMessage = "";
    }
}

//TEST
for(numberOfTraces=0; numberOfTraces<9; numberOfTraces++){
    System.out.println(traces[numberOfTraces]);
}
```

```
// GET IMAGE WITH TRACES

File imageWithTraces = new File("C:\\Users\\Evripidis\\Desktop\\imageWithTraces1.jpeg");
FileOutputStream fileWriter1 = new FileOutputStream(imageWithTraces);
try{
    outMessage="P7462";
    for(int i=0; i<9; i++){
        outMessage +="T=" + traces[i];
    }
    outMessage+="\r";
    System.out.print(outMessage);
    modem.write(outMessage.getBytes());
    byte previous=(byte)modem.read();
    byte input=(byte) modem.read();
    System.out.print((char)previous+"\n");
    int reps=0;
    while(!(previous==(byte) 0xFF && input==(byte) 0xD8) && reps<10000){
        previous=(byte)input;
        //System.out.print((char)previous+"\n");
        input=(byte)modem.read();
        reps++;
    }
    if(reps==10000){
        System.out.print("ERROR\n");
    }
    else{
        imageWithTraces.createNewFile();
        fileWriter1.write(previous);
        fileWriter1.write(input);
        System.out.print("Start of image delimeter 0xFF\n");
        System.out.print("Start of image delimeter 0xD8\n");
        boolean flag=false;
        while(flag==false){
            input=(byte) modem.read();
            fileWriter1.write(input);
            if(input==(byte)0xD9 && previous==(byte)0xFF){
                System.out.print("End of image delimeter 0xFF\n");
                System.out.print("End of image delimeter 0xD9\n");
                System.out.print("END OF IMAGE\n");
                fileWriter1.close();
                flag=true;
                break;
            }
            previous=(byte)input;
        }
    }
}
```

```
            catch  (IOException io) {
                io.printStackTrace();
            }
        }
        catch (FileNotFoundException f) {
            f.printStackTrace();
        }
        System.out.print("Image with trace sent.\n");
    }

    public void ARQ(Modem modem) {
        File graph3 = new File("C:\\Users\\Evripidis\\Desktop\\graph31.txt");
        boolean flag=true;
        long currentTime;
        long response;
        int numOfPackets = 0;
        int wrongs = 0;
        int totwrongs = 0;
        try {
            FileWriter fileWriter = new FileWriter(graph3);
            long startTime=System.currentTimeMillis();
            while(System.currentTimeMillis()<=startTime+240000) {
                System.out.println("MPHKA");
                if(flag) {
                    outMessage="Q6979\r";
                    modem.write(outMessage.getBytes());
                }
                else {
                    outMessage="R0974\r";
                    modem.write(outMessage.getBytes());
                }
                currentTime=System.currentTimeMillis();
                inMessage=null;
                for(;;) {
                    int input=modem.read();
                    inMessage+=(char)input;
                    System.out.print((char)input);
                    if(inMessage.contains(pstop)){
                        break;
                    }
                }
                numOfPackets++;
                System.out.print("Package read. \n");
                String crypto=inMessage.substring(35, 51);
                String fcs=inMessage.substring(53, 56);
                System.out.print(crypto+"\n");
                System.out.print(fcs+"\n");
                byte[] pinakas=crypto.getBytes();
                byte apotelesma=pinakas[0];
```

```
                for (int i=1;i<16;i++){
                apotelesma= (byte) (apotelesma ^ ((byte) (pinakas[i])));
                }
                int fcs1=Integer.parseInt(fcs);
                System.out.print(fcs1+"\n");
                System.out.print((int)apotelesma+"\n");
                if(fcs1==(int)apotelesma){
                    flag=true;
                    response=System.currentTimeMillis()-currentTime;
                    String content = String.valueOf(response);
                    fileWriter.write(content+ " " +wrongs+"\r\n");
                    System.out.print("ACK\n");
                    wrongs = 0;
                }
                else {
                    flag=false;
                    //response=System.currentTimeMillis()-currentTime;
                    //String content = String.valueOf(response);
                    //fileWriter.write(content+" 1"+"\r\n");
                    System.out.print("NACK\n");
                    wrongs++;
                    totwrongs++;
                }
            }
            fileWriter.close();
            System.out.print("ARQ terminated.\n");
        }
        catch  (IOException io) {
            io.printStackTrace();
        }
        double BER = 1 - Math.pow((double)totwrongs/numOfPackets, (double)1/16);
        System.out.println("BER equals: " +BER);
    }
}
```