package UserApplication;

import java.lang.\*;

import ithakimodem.Modem;

import java.io.\*;

import java.lang.System;

import java.util.Date;

import java.util.concurrent.TimeUnit;

import java.io.IOException;

import java.io.File;

import java.lang.Math;

import java.io.PrintWriter;

public class virtualModem {

public static void main(String[] param) throws IOException {

(new virtualModem()).demo();

}

public void demo() throws IOException {

int k;

boolean b;

Modem modem;

modem = new Modem();

modem.setSpeed(80000);

modem.setTimeout(2000);

modem.open("ithaki");

byte[] echo = "EXXXX\r".getBytes();

byte[] imgErF = "MXXXX\r".getBytes();

byte[] imgEr = "GXXXX\r".getBytes();

byte[] gps = "PXXXX=1000199\r".getBytes();

byte[] ack = "QXXXX\r".getBytes();

byte[] nack = "RXXXX\r".getBytes();

PrintWriter responseTimesEcho = new PrintWriter("responseTimesEcho.txt", "UTF-8");

FileOutputStream imageErrorFree = new FileOutputStream("E1.jpg");

FileOutputStream imageError = new FileOutputStream("E2.jpg");

FileOutputStream imageGPS = new FileOutputStream("M1.jpg");

for (int i = 0; i < 205; i++) {

try {

k = modem.read();

if (k == -1)

break;

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

} catch (Exception x) {

break;

}

}

long timeRun = System.currentTimeMillis();

long endRun = timeRun + 300000;

// echo packets

while (System.currentTimeMillis() < endRun) {

responseTimesEcho.println(echoPackets(echo, modem));

}

responseTimesEcho.close();

// images

images(imgErF, modem, imageErrorFree);

imageErrorFree.close();

images(imgEr, modem, imageError);

imageError.close();

// GPS

int map[][] = new int[99][80];

map = gps(modem, gps);

long T[] = new long[5];

T = gpsT(map);

String codeForPins = "PXXXX=" + T[4];

for (int i = 0; i < 4; i++) {

codeForPins += "T=" + T[3 - i];

}

codeForPins += "\r";

System.out.println(codeForPins);

byte[] gpsPins = codeForPins.getBytes();

images(gpsPins, modem, imageGPS);

imageGPS.close();

{

try {

k = modem.read();

if (k == -1)

break;

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

} catch (Exception x) {

break;

}

}

// ARQ

long ArqTime = System.currentTimeMillis();

long ArqEndTime = ArqTime + 300000;

PrintWriter responseTimesForArq = new PrintWriter("responseArq.txt", "UTF-8");

PrintWriter NumOfRetransmissions = new PrintWriter("retransmissions.txt", "UTF-8");

int errors = 0;

int numOfpackets = 0;

int[] stats = new int[2];

while (System.currentTimeMillis() < ArqEndTime) {

stats = arqFunc(modem, ack, nack, responseTimesForArq, NumOfRetransmissions);

errors += stats[1];

numOfpackets += stats[0];

}

System.out.println(errors);

System.out.println(numOfpackets);

responseTimesForArq.close();

NumOfRetransmissions.close();

modem.close();

}

public int echoPackets(byte[] echoReq, Modem modem1) {

modem1.write(echoReq);

int k;

int counter = 0;

long startT = System.currentTimeMillis();

long endT = 0;

long responseT;

for (counter = 0; counter < 35; counter++) {

try {

k = modem1.read();

if (k == -1) {

break;

}

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

} catch (Exception x) {

break;

}

}

endT = System.currentTimeMillis();

responseT = endT - startT;

return (int) responseT;

}

public void images(byte[] imageErFr, Modem m, FileOutputStream imgErrF) throws IOException {

m.write(imageErFr);

//μεταβλητες για τον ελεγχο του τελους της εικονας

int last = 0;

int preLast = 0;

int n = 0;

while (true) {

preLast = last;

last = m.read();

if (last == -1)

break;

if (preLast == 255 && last == 216) {

n = 1;//αλλαγη της τιμης της n

imgErrF.write(255);

imgErrF.write(216);

while (true) {

try {

last = m.read();

if (preLast == 255 && last == 217) {

imgErrF.write((byte) last);

break;

}

preLast = last;

imgErrF.write((byte) last);

if (last == -1)

break;

}

catch (Exception x) {

break;

}

}

if (n == 1)

break;

}

}

imgErrF.close();

}

public void gps(){

String GPS = new String();

int e;

for(;;){

try{

e=modem.read();

if(e==-1)break;

}catch(Exception x){

System.exit(1);

}

}

OutputStream op=null;

PrintWriter coordinates=null;

boolean counter=false;

OutputStream out=null;

int k;

try{

out=new FileOutputStream("M1.jpg");

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

try{

op=modem.getOutputStream();

}catch (Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

try{

coordinates=new PrintWriter("Coordinates.txt","UTF-8");

}catch(Exception x){

System.out.println("Creation Error");

System.exit(1);

}

try{

op.write(GPS.getBytes());

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

for(;;){

try{

k=modem.read();

if(k==-1)break;

coordinates.write((char)k);

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

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

}

coordinates.close();

try{

op.close();

modem.close();

}catch(Exception x){

System.out.println("Exception Occured ");

System.exit(1);

}

String line="";

BufferedReader reader1=null;

PrintWriter onlyCoordinates=null;

try{

onlyCoordinates=new

PrintWriter("onlyCoordinates.txt","UTF-8");

}catch(Exception x){

System.out.println("Exception Occured");

System.exit(1);

}

try{

reader1=new BufferedReader(new

FileReader("Coordinates.txt"));

}catch (Exception x){

System.out.println("Exception Occured");

System.exit(1);

}

try{

while((line=reader1.readLine())!=null){

if(line.startsWith("$GPGGA")){

onlyCoordinates.write(line);

onlyCoordinates.write(System.lineSeparator());

}

}

}catch(Exception x){

System.exit(1);

}

try{

reader1.close();

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

try{

onlyCoordinates.close();

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

BufferedReader reader2=null;

line="";

try{

reader2=new BufferedReader(new

FileReader("onlyGPS.txt"));

}catch(Exception x){

System.exit(1);

}

int Counter=0;

double[] longtitude=new double[5];

double[] latitude=new double[5];

int[] time=new int[5];

int sec,min,temp,temp2;

String TEMP;

String[][] data=new String[99][15];

try{

while((line=reader2.readLine())!=null){

if(Counter==99)break;

data[Counter]=line.split(",");

Counter=Counter+1;

}

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

try{

reader2.close();

}catch(Exception x){

System.exit(1);

}

Counter=0;

for(int j=0;j<data.length;j++){

TEMP=data[j][1].substring(2,6);

temp=Integer.parseInt(TEMP);

sec=temp%100;

min=(temp%10000)-sec;

min=min/100;

temp2=(min\*60);

temp2+=sec;//temp2 time in sec

if(Counter==0){

latitude[Counter]=Double.parseDouble(data[j][2]);

longtitude[Counter]=Double.parseDouble(data[j][4]);

time[Counter]=temp2;

Counter+=1;

}

else if(Counter<5 && Counter>0){

if(temp2-time[Counter-1]>18){

latitude[Counter]=Double.parseDouble(data[j][2]);

longtitude[Counter]=Double.parseDouble(data[j][4]);

time[Counter]=temp2;

Counter=Counter+1;

}

}

else break;

}

String cmd=GPS;

long a,b;

int aa,bb;

for(int j=0;j<5;j++){

a=(long)(longtitude[j]);

b=(long)(latitude[j]);

aa=(int)((longtitude[j]-a)\*60);

bb=(int)((latitude[j]-b)\*60);

cmd=cmd+"T="+a+aa+b+bb;

}

cmd=cmd+"\r\n";

System.out.println(cmd);

Modem modem2=new Modem();

modem2.setSpeed(80000);

modem2.setTimeout(2000);

modem2.open("ithaki");

for(;;){

try{

k=modem2.read();

if(k==-1)break;

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

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

}

for(;;){

try{

modem2.write(cmd.getBytes());

}catch(Exception x){

System.out.println("Raise Exception");

System.exit(1);

}

try{

k=modem2.read();

if(k==-1)break;

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

if(k==0xFF){

for(;;){

out.write(k);

k=modem2.read();

if(k==0xFF){

out.write(k);

k=modem2.read();

if(k==0xD9){

out.write(k);

counter=true;

}

}

if(counter)break;

}

}

}catch(Exception x){

System.exit(1);

}

if(counter)break;

}

modem2.close();

try{

out.close();

}catch(Exception x){

System.exit(1);

}

}

public int[] arqFunc(Modem modem1, byte[] ack\_code, byte[] nack\_code, PrintWriter txt1, PrintWriter txt2) {

modem1.write(ack\_code);

int k = 0;

int counter = 0;

long startT = System.currentTimeMillis();// χρονος εκκινησης του προγραμματος

long endT = 0;

long responseT;

int arrayPck[] = new int[58];

int seq[] = new int[16];

int valueOfXor;

int valueOfFcs;

int[] stats = new int[2];

for (;;) {

try {

for (counter = 0; counter < 58; counter++) {

k = modem1.read();

arrayPck[counter] = k; // τοποθετηση του πακετου σε εναν πινακα

}

for (int j = 0; j < 16; j++) {

seq[j] = arrayPck[31 + j];// σε αυτο το σημειο παιρνουμε τον 16αδικο κωδικο

}

valueOfXor = seq[0] ^ seq[1];

for (int l = 0; l < 14; l++) {

valueOfXor = valueOfXor ^ seq[l + 2];

}

String str1;

str1 = "";

for (int i = 0; i < 3; i++) {

str1 += Integer.toString(arrayPck[49 + i] - 48);

}

valueOfFcs = Integer.parseInt(str1);

if (valueOfFcs == valueOfXor) {

for (int j = 0; j < 58; j++) {

System.out.print((char) arrayPck[j]);

}

// test

stats[0]++;

System.out.println();

System.out.println("valueOfFcs = " + valueOfFcs + "valueOfXor = " + valueOfXor);

break;

}

else {

modem1.write(nack\_code);

stats[1]++;

System.out.println(" error! wrong" + "valueOfFcs = " + valueOfFcs + "valueOfXor = " + valueOfXor);

}

if (k == -1) {

break;

}

} catch (Exception x) {

break;

}

}

endT = System.currentTimeMillis();

responseT = endT - startT;

txt1.println(responseT);

txt2.println(stats[1]);

return stats;

}

}