```
package userApplication;

import java.net.*;

import java.util.ArrayList;


import javax.sound.sampled.AudioFormat;

import javax.sound.sampled.AudioSystem;

import javax.sound.sampled.LineUnavailableException;

import javax.sound.sampled.SourceDataLine;


import java.io.*;

import java.lang.System;


public class userApplication {


  public static void main(String[] args) throws IOException, LineUnavailableException {

  userApplication app = new userApplication();


  //κωδικοι ιθακης


  String echo = "E2624";

  String echo2 = "E0000";

  String imgCode = "M2355FLOW=ON";

  String imgCode2 = "M2355CAM=PTZFLOW=ON";

  String audio = "A2201L20F999";

  String frequency = "A2201T999";
  //String obdii = "V9877OBD=01 0C";


  int serverPort = 38020;

  int clientPort = 48020;


 //ηχος

  //System.out.println("ΜΡΗΚΑ ΗΧΟ");

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

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


  PrintWriter samplesAQ1= new PrintWriter("samplesAQ1.txt","UTF-8");

  PrintWriter samplesAQ2= new PrintWriter("samplesAQ2.txt","UTF-8");


  PrintWriter diffsAQ1= new PrintWriter("diffsAQ1.txt","UTF-8");

  PrintWriter diffsAQ2= new PrintWriter("diffsAQ2.txt","UTF-8");


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

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


  app.getAudio(audio,serverPort,clientPort,960,"DPCM",samplesDPCM,diffDPCM);//DPCM

  app.getAudio("A2201AQL11F999",serverPort,clientPort,960,"AQ-DPCM",samplesAQ1,diffsAQ1);//AQ

  app.getAudio("A2201AQL43F999",serverPort,clientPort,960,"AQ-DPCM",samplesAQ2,diffsAQ2);//AQ

  app.getAudio(frequency,serverPort,clientPort,960,"DPCM",samplesFreq,diffFreq);// γεννητρια


  samplesDPCM.close();
```

```
    diffDPCM.close();

    samplesAQ1.close();

    samplesAQ2.close();

    diffsAQ1.close();

    diffsAQ2.close();

    samplesFreq.close();

    diffFreq.close();


//ithakiCopter


    app.ithakiCopter(38048,48038);


    //echo κληση στη main

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

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


    app.echo(echo,serverPort,clientPort,echoResponseTimesTXT,throughput);


    throughput.close();

    echoResponseTimesTXT.close();


    //echo χωρις delay

    PrintWriter echoResponseTimesTXT2 = new PrintWriter("responseTimes2.txt","UTF-8");

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

```
app.echo(echo2,serverPort,clientPort,echoResponseTimesTXT2,throughputNoDelay);


throughputNoDelay.close();

echoResponseTimesTXT2.close();


//θερμοκρασιες

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


app.temp(echo,serverPort,clientPort,temps);


temps.close();


//εικονα


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

FileOutputStream imageOut2=new FileOutputStream("E2.jpg");

app.getImage(imgCode,serverPort,clientPort,imageOut);

app.getImage(imgCode2,serverPort,clientPort,imageOut2);

imageOut.close();

imageOut2.close();




//obd

app.obdii("V2477OBD=01 1F",serverPort,clientPort,"1F");
```

```
    app.obdii("V2477OBD=01 0F",serverPort,clientPort,"0F");

    app.obdii("V2477OBD=01 11",serverPort,clientPort,"11");

    app.obdii("V2477OBD=01 0C",serverPort,clientPort,"0C");

    app.obdii("V2477OBD=01 0D",serverPort,clientPort,"0D");

    app.obdii("V2477OBD=01 05",serverPort,clientPort,"05");


    }




 //ΣΥΝΑΡΤΗΣΕΙΣ ΠΟΥ ΧΡΗΣΙΜΟΠΟΙΩ ΣΤΗ MAIN------------------------------------------------------------------------
-------------------------------


public void requestCode(String cd, int serverPort) throws IOException { //κανει ενα request στην ιθακη
για ενα πακετο που θα καθοριστει απο τον κωδικο που θα της βαλω σαν ορισμα


    byte[] code = cd.getBytes();//bytes του κωδικου της ιθακης


    byte[] hostIP = { (byte)155,(byte)207,(byte)18,(byte)208 };//Ip ιθακης

    InetAddress hostAddress = InetAddress.getByAddress(hostIP);


    DatagramSocket s = new DatagramSocket(); //δημιουργει socket send για να στειλω στον σερβερ τον
κωδικο του αιτηματος


    DatagramPacket p = new DatagramPacket(code,code.length,hostAddress,serverPort);


    s.send(p);
```

```
   s.close();


}


public void echo(String echoCode,int serverPort, int clientPort,PrintWriter txt,PrintWriter throughput)
throws IOException {


            ArrayList<Integer> latencies = new ArrayList<Integer>();


            if(echoCode=="E0000") {throughput = new PrintWriter("throughputNoDelay.txt","UTF-8");}

            else {throughput = new PrintWriter("throughput.txt","UTF-8");}


            long startTime=0;

   long endTime=0;

   int responseTime=0;

   int packetsCounter=0;


   long timeRun=0;


   byte[] rxbuffer = new byte[32]; //πινακας που θα δεχτει το response της ιθακης σε Bytes


   DatagramSocket r = new DatagramSocket(clientPort); //πυλη για να ερθει το response της ιθακης

   DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length); //εδω θα ερθει το response της
ιθακης
```

```
r.setSoTimeout(2500);

timeRun = (System.currentTimeMillis())+(1000*60*4);

 while (System.currentTimeMillis()<=timeRun) { //for για 4 λεπτα

           requestCode(echoCode,serverPort);

           try {


         startTime=System.currentTimeMillis();//--ξεκιναει το πρωτο πακετο


         r.receive(q);


         endTime=System.currentTimeMillis();//--εφτασε το πρωτο πακετο


         responseTime=(int)(endTime-startTime);


         latencies.add(responseTime);


         String message = new String(rxbuffer,0,q.getLength());

         System.out.println(message);

         packetsCounter++;

         txt.println(responseTime);


     } catch (Exception x) {

         System.out.println(x);

     }
```

```
       }
     txt.println();
     txt.println();
     txt.println();
     txt.println("Arithmos paketvn"+ " " + packetsCounter);

     throughput(latencies,throughput);

     throughput.close();
     r.close();

}

public void temp(String echoCode,int serverPort, int clientPort,PrintWriter txt) throws IOException {
   byte[] rxbuffer = new byte[54]; //πινακας που θα δεχτει το response της ιθακης σε Bytes

   DatagramSocket r = new DatagramSocket(clientPort); //πυλη για να ερθει το response της ιθακης
   DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length); //εδω θα ερθει το response της
ιθακης

   r.setSoTimeout(2500);

   String kwdikos=echoCode+"T00";
```

```
    requestCode(kwdikos,serverPort);


    r.receive(q);


    String temperature = new String(rxbuffer,43,3);

    System.out.println("ΜΡΗΚΑ TEMP");

    txt.println("Temp is "+ temperature );


    String message = new String(rxbuffer,0,q.getLength());

    System.out.println(message);


r.close();

}

public void getImage(String imgCode,int serverPort, int clientPort,FileOutputStream img) throws
IOException {


    byte[] rxbuffer = new byte[128]; //πινακας που θα δεχτει το response της ιθακης σε Bytes


    DatagramSocket r = new DatagramSocket(clientPort); //πυλη για να ερθει το response της ιθακης

    DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length); //εδω θα ερθει το response της
ιθακης




    //r.setSoTimeout(1400); //timeout της πυλης

    int o=0;

    int startControl=0;
    while(o==0) {// μεσα σε αυτη τη λουπα παιρνουμε ενα ενα τα πακετα των 128 και τα γραφουμε
στον Buffer, μετα τα γραφουμε στο αρχειο img μας και παιρνουμε το επομενο πακετο μεχρι να βρουμε
κωδικο τελους

        requestCode(imgCode,serverPort);

        r.receive(q);

        //String message = new String(rxbuffer,0,q.getLength());

        //System.out.println(message);


        if(startControl==0){//ψαχνει μεχρι να βρει την αρχη της εικονας, οταν την βρει γραφει απο
εκεινο το σημειο κι επειτα τα bytes στο αρχειο και κανει ενα receive ωστε να προχωρησει στο αμεσως
επομενο βημα η συναρτηση

        for(int i=0;i<128;i++) {

         if(i<127)if(rxbuffer[i]==(byte)0xFF && rxbuffer[i+1]==(byte)0xD8)

         {

                startControl=1;

                for(int l=i; l<128; l++) {

              img.write(rxbuffer[l]);}

                r.receive(q);

         }

        }}


        if(startControl==1) {




          for(int i=0; i<128; i++) {

            img.write(rxbuffer[i]);

            if(i<127)if(rxbuffer[i]==(byte)0xFF && rxbuffer[i+1]==(byte)0xD9)
```

```
                {
                img.write(rxbuffer[i+1]);
                break;
                }


    }
         for(int i=0;i<127;i++)if(rxbuffer[i]==(byte)0xFF && rxbuffer[i+1]==(byte)0xD9) {
            o=1;
            System.out.println(rxbuffer[i]);
            System.out.println(rxbuffer[i+1]);
            }   }
    }


    r.close();
}


public void getAudio(String audioCode,int serverPort, int clientPort,int packetNum,String
enc,PrintWriter samples, PrintWriter diffs) throws IOException, LineUnavailableException {
        //μεταβλητες που θα χρειαστω παρακατω
        int packetSize;
        if(enc=="DPCM")packetSize=128;
        else packetSize=132;
        int bit;
        if(enc=="DPCM")bit=8;
        else bit=16;
        int arrayMultiplier;
```

```
if(enc=="DPCM")arrayMultiplier=1;

else arrayMultiplier=2;



AudioFormat linearPCM = new AudioFormat(8000,bit,1,true,false);


byte[] rxbuffer = new byte[packetSize];//999 γτ ζηταω τοσα πακετα αρχικα

byte[][] audioBuffer= new byte[999][packetSize];

byte[][] audioBufferS=new byte [packetNum][256*arrayMultiplier];

byte[] audioBufferOut = new byte[256*arrayMultiplier*packetNum];


int OOOOIIII=15; // για το nibble 2

int IIIIOOOO=240;// για το nibble 1


int nibble1,diff1; // [ αυτο    |        ]

int nibble2,diff2; // [        |  αυτο   ]


int sample0=0;  //[ αυτο    |        ]

int sample1;//[        |  αυτο   ]

int sample2;


int beta=1;


DatagramSocket r = new DatagramSocket(clientPort);

DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);
```

```
        // τελος μεταβλητων


        r.setSoTimeout(2500);

        requestCode(audioCode,serverPort);

        //for για να γεμισω τον πινακα audioBuffer


        for(int i=0; i<999;i++) {//999 receive κι οσα παρει


        try{r.receive(q);}

        catch(Exception x) {break;}

                System.out.println(i);

                //String message = new String(rxbuffer,0,q.getLength());

    System.out.println((int)rxbuffer[0]);

        for(int o=0;o<rxbuffer.length;o++) {

        audioBuffer[i][o]=rxbuffer[o];

        }

        }//τελος της for

        r.close();


        if(enc=="DPCM") {


        for(int k=0;k<packetNum;k++) {

                for(int a=0;a<packetSize;a++) {

                        nibble1=(audioBuffer[k][a]) & IIIIOOOO;

                        nibble2=(audioBuffer[k][a]) & OOOOIIII;
                        diff1=((nibble1>>4)-8)*beta;

                        diff2=(nibble2-8)*beta;


                        sample1=diff1+sample0;

                        sample2=diff2+sample1;

                        sample0=sample2;


                        samples.println(sample1);

                        samples.println(sample2);


                        diffs.println(diff1);

                        diffs.println(diff2);


                        audioBufferS[k][a*2]=(byte)sample1;

                        audioBufferS[k][a*2+1]=(byte)sample2;

                        System.out.println((int)audioBufferS[k][a*2]);

                }

                }}


        int mlsb,mmsb,blsb,bmsb;

        int mean;

        int b;
        if(enc=="AQ-DPCM") {


                PrintWriter means = new PrintWriter("mean"+audioCode+".txt","UTF-8");

                PrintWriter steps = new PrintWriter("step"+audioCode+".txt","UTF-8");


                int sample1_1;

                int sample1_2;

                int sample2_1;

                int sample2_2;


                for(int a=0; a<packetNum;a++) {

                        mlsb=audioBuffer[a][0]&0xFF;

                        mmsb=audioBuffer[a][1]&0xFF;

                        blsb=audioBuffer[a][2]&0xFF;

                        bmsb=audioBuffer[a][3]&0xFF;


                        mean=(mmsb<<8)| mlsb;

                        b=(bmsb<<8)| blsb;


                        means.println(mean);

                        steps.println(b);



                        for(int k=4;k<packetSize;k++) {

                                nibble1=(audioBuffer[a][k]) & IIIIOOOO;
```

```
                    nibble2=(audioBuffer[a][k]) & OOOOIIII;


                    sample1=((nibble1>>4)-8)*b+mean;

                    sample2=(nibble2-8)*b+mean;


                    sample1_1=(sample1 & 255);

                    sample1_2=((sample1 & 65280)>>8);


                    sample2_1=(sample2 & 255);

                    sample2_2=((sample2 & 65280)>>8);


                    samples.println(sample1_1);

                    samples.println(sample1_2);

                    samples.println(sample2_1);

                    samples.println(sample2_2);


                    diffs.println((nibble1>>4)-8);

                    diffs.println(nibble2-8);




                    audioBufferS[a][(k-4)*4]=(byte)sample1_1;

                    audioBufferS[a][(k-4)*4+1]=(byte)sample1_2;

                    audioBufferS[a][(k-4)*4+2]=(byte)sample2_1;

                    audioBufferS[a][(k-4)*4+3]=(byte)sample2_2;

                }}
```

```
        means.close();

        steps.close();}



    SourceDataLine lineOut = AudioSystem.getSourceDataLine(linearPCM);


    lineOut.open(linearPCM,256*arrayMultiplier*packetNum);


    lineOut.start();


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

        for(int k=0;k<256*arrayMultiplier;k++) {

        audioBufferOut[l*256*arrayMultiplier+k]=audioBufferS[l][k];


        }

    }





    lineOut.write(audioBufferOut,0,256*arrayMultiplier*packetNum);


    lineOut.stop();


    lineOut.close();
```

```
}

public void obdii(String obdCode, int serverPort,int clientPort, String pid) throws IOException {
            byte[] rxbuffer = new byte[128];

            PrintWriter vehicleDiag = new PrintWriter("vehicleDiag"+pid+".txt","UTF-8");

   DatagramSocket r = new DatagramSocket(clientPort);
   DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);

   r.setSoTimeout(2500);

   long timeRun = (System.currentTimeMillis())+(1000*60*4);

   while(System.currentTimeMillis()<=timeRun) {

   requestCode(obdCode,serverPort);
  try {
   r.receive(q);
  }catch(Exception x) {continue;}
   String message = new String (rxbuffer,0,q.getLength());
   System.out.println(message);
   //txt.println()


   int dec1;

   //int dec2;


   switch(pid) {

           case "1F":


                       dec1=(256* (Integer.parseInt(message.substring(6,8),16)) +
Integer.parseInt(message.substring(9,11),16));

                       vehicleDiag.println(dec1);


                       break;

           case "0F":


                       dec1=(Integer.parseInt(message.substring(6,8),16))-40;

                       vehicleDiag.println(dec1);

                       break;


           case "11":

                       dec1=((Integer.parseInt(message.substring(6,8),16))*100)/255;

                       vehicleDiag.println(dec1);

                       break;

           case "0C":

                       dec1=((256* (Integer.parseInt(message.substring(6,8),16)) +
Integer.parseInt(message.substring(9,11),16)))/4;
```

```
                    vehicleDiag.println(dec1);

                    break;

          case "0D":

                    dec1=Integer.parseInt(message.substring(6,8),16);

                    vehicleDiag.println(dec1);

                    break;

          case "05":

                    dec1=Integer.parseInt(message.substring(6,8),16)-40;

                    vehicleDiag.println(dec1);

                    break;

}


   }




   vehicleDiag.close();


          r.close();


}



public void tcpReq() throws IOException {

/////TCP socket
```

```
        Socket s = new Socket();


        InetAddress addr = InetAddress.getByName("ithaki.eng.auth.gr");

        SocketAddress sa = new InetSocketAddress(addr, 38048);


        s.connect(sa);


        OutputStream out =s.getOutputStream();


////TCP socket

        String req = "AUTO FLIGHTLEVEL=150 LMOTOR=010 RMOTOR=010 PILOT";

        byte[] c = req.getBytes();


        for(int i=0;i<47;i++) {


                out.write(c);


        }


        s.close();


}


public void ithakiCopter(int serverPort,int clientPort) throws IOException {
        PrintWriter ithakiCopter_l = new PrintWriter("ithakiCopter_l.txt","UTF-8");

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

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

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

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


        String left,right,altitude,temp,pressure;


        byte[] rxbuffer = new byte[128];


  DatagramSocket r = new DatagramSocket(clientPort);

  DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);

  //tcpReq();

  r.setSoTimeout(2500);


  for(int i=0;i<120;i++) {

  try {

  r.receive(q);}

  catch(Exception x) {break;}


  left = new String(rxbuffer,40,3);

  right = new String(rxbuffer,51,3);

  altitude = new String(rxbuffer,64,4);

  temp = new String(rxbuffer,80,7);

  pressure = new String(rxbuffer,96,8);
```

```
    ithakiCopter_l.println(left);

    ithakiCopter_r.println(right);

    ithakiCopter_altitude.println(altitude);

    ithakiCopter_temp.println(temp);

    ithakiCopter_pressure.println(pressure);


    String message = new String (rxbuffer,0,q.getLength());

    System.out.println(message + "---" + left+ "---" +right+ "---" + altitude + "---" + temp + "---" + pressure+
"---" );

    }


    ithakiCopter_l.close();

    ithakiCopter_r.close();

    ithakiCopter_altitude.close();

    ithakiCopter_temp.close();

    ithakiCopter_pressure.close();


            r.close();




}


public void throughput(ArrayList<Integer> latencyArr,PrintWriter throughput) {


            int startTime;
```

```
        //long memory;

        int extra=0;


        int sum=0; //μεχρι 8000ms

        float packetCount=0;


        for(int time=0;time<232;time++) {//240=60*4mins - 8 sec


                startTime=time*1000;


        for(int i=0;i<latencyArr.size();i++) {

                if(startTime>0) {

                if(latencyArr.get(i)<startTime) {

                        startTime=startTime-latencyArr.get(i);

                        continue;


                }

                if(latencyArr.get(i)>startTime) {

                        sum=latencyArr.get(i)-startTime;

                        startTime=0;

                        packetCount=(float)sum/(float)latencyArr.get(i);//παιρνω το division του
πακετου

                        continue;


                }}

                sum=sum+latencyArr.get(i);
                packetCount++;

                if(sum>8000) {

                        extra=sum-8000;

                        packetCount--;

                        packetCount=packetCount+(float)(latencyArr.get(i)-
extra)/(float)latencyArr.get(i);

                        throughput.println((float)(packetCount/(float)8));

                        break;

                }




        }

        sum=0;

        packetCount=0;


        }


}
```

```
}
```