Source

USER APPLICATION:

```
import java.io.IOException;
import java.net.InetAddress;
import java.util.Scanner;
import javax.sound.sampled.LineUnavailableException;

public class userApplication {

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

		Scanner scn = new Scanner(System.in);
		String packetInfo;
		int nP = 999; // Number Of Packets
		int serverPort = 38000, clientPort = 48000, mode, run=0;
		byte[] hostIP = { (byte)155, (byte)207, (byte)18, (byte)208 };
		InetAddress hostAddress = InetAddress.getByAddress(hostIP);

//-------------------------------------SCREEN_MESSAGES / SELECT OPTION-------
-------------------------
		System.out.print("OPTIONS:\n1. Echo Packet\n2. Image\n3. Audio
(DPCM)"
				+ "\n4. Audio (AQDPCM)\n5. Ithaki Copter (TCP)\n6.
Ithaki Copter (UDP)"
				+ "\n7. Car Fault Diagnostics (UDP)\n\nENTER NUMBER:
");
		int option = scn.nextInt();

//-----------------------------------------------OPTION_RUN-----------------
-------------------------
		switch(option) {
			case 1:	//-----------------------------------------
---------------------------ECHO_PACKET
				packetInfo = "EXXXX";
				System.out.print("\n1. Response Time -
Throughputs\n2. Temperatures\n\nENTER NUMBER: ");
				mode = scn.nextInt();
				if(mode == 2) {
					packetInfo = packetInfo + "T";
				}else {
					System.out.print("\n1. With Delay\n2. Without
Delay\n\nENTER NUMBER: ");
					run = scn.nextInt();
					if (run == 2)
						packetInfo = "E0000";
				}
				new echoPacket().run(packetInfo, serverPort,
clientPort, hostAddress, mode, run);
				break;
```

```
                    case 2:       //-----------------------------------------
--------------------------------IMAGE
                                packetInfo = "MXXXX";
                                System.out.print("\nWhich Camera do you want to
use?\n1. Camera 1 \n2. Camera 2"
                                                + "\n\nENTER NUMBER: ");
                                int cam = scn.nextInt();
                                if (cam == 2)
                                        packetInfo = packetInfo+"CAM=PTZ";
                                new image().run(packetInfo, serverPort, clientPort,
hostAddress, cam);
                                break;
                    case 3:       //-----------------------------------------
---------------------------AUDIO(DPCM)
                                packetInfo = "AXXXX";
                                System.out.print("\n1. Frequencies\n2.
Audio\n\nENTER NUMBER: ");
                                mode = scn.nextInt();
                                if (mode == 1)
                                        packetInfo = packetInfo + "T" + nP;
                                else
                                        packetInfo = packetInfo + "F" + nP;
                                new audioDPCM().run(packetInfo, serverPort,
clientPort, hostAddress, nP);
                                break;
                    case 4:       //-----------------------------------------
---------------------------AUDIO(AQDPCM)
                                packetInfo = "AXXXXAQF"+nP;
                                new audioAQDPCM().run(packetInfo, serverPort,
clientPort, hostAddress, nP);
                                break;
                    case 5:       //-----------------------------------------
--------------------ITHAKI_COPTER_TCP
                                int portNumber = 38048;
                                String request = "AUTO FLIGHTLEVEL=150 LMOTOR=150
RMOTOR=150 PILOT \r\n";
                                new ithakiCopter_I().run(portNumber, hostAddress,
request);
                                break;
                    case 6:       //-----------------------------------------
--------------------ITHAKI_COPTER_UDP
                                packetInfo = "QXXXX";
                                serverPort = 38078;
                                clientPort = 48078;
                                new ithakiCopter_II().run(packetInfo, serverPort,
clientPort, hostAddress);
                                break;
                    case 7:       //-----------------------------------------
-------------CAR_FAULT_DIAGNOSTICS_UDP
                                packetInfo = "VXXXX";
                                new carFaultDiag_II().run(packetInfo, serverPort,
clientPort, hostAddress);
                                break;
            }
            scn.close();
    }
```

ECHO PACKET:

```
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;

public class echoPacket {

	ArrayList<Long> responseTime = new ArrayList<Long>();
	ArrayList<Long> throughputBPS = new ArrayList<Long>();
	ArrayList<Long> srttAr = new ArrayList<Long>(), varAr = new
ArrayList<Long>();
	ArrayList<Long> rtoAr = new ArrayList<Long>();
	ArrayList<String> messages = new ArrayList<String>();
	int ack=0, nack=0, counter = 10, packets = 0;
	long srtt=0, var=0, totalTime = 0;

	public void run(String pInfo, int sPort, int cPort, InetAddress hostAdr,
int mode, int run) throws IOException {

		DatagramSocket s = new DatagramSocket();
		byte[] txbuffer = pInfo.getBytes();
		DatagramPacket p = new
DatagramPacket(txbuffer,txbuffer.length,hostAdr,sPort);
		DatagramSocket r = new DatagramSocket(cPort);
		r.setSoTimeout(4000);
		byte[] rxbuffer = new byte[2048];
		DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);
		if (run == 2)
			r.setSoTimeout(1000);

		if (mode == 1) {
			long strTime = System.currentTimeMillis();
			while (System.currentTimeMillis() < (strTime+300000)) {
				try {
					long startTime = System.currentTimeMillis();
					s.send(p);
					r.receive(q);
					ack++;
					long response = System.currentTimeMillis() -
startTime;
					totalTime += response;
					packets++;
					srtt = (long)((0.9*srtt) + ((1-
0.9)*response));//------------------- a = 0.9
					var = (long)((0.25*var) + Math.abs(((1-
0.25)*(srtt-response))));//-- b = 0.25
					long rto = srtt + 4*var;//------------------
--------------------------- c = 4
					responseTime.add(response);
					srttAr.add(srtt);
					varAr.add(var);
					rtoAr.add(rto);
```

```
                                    String message = new
String(rxbuffer,0,q.getLength());
                                    System.out.println(message);
                                    if (totalTime >= 8000) {

      throughputBPS.add((long)(packets*8*32)/8);
                                          totalTime = 0;
                                          packets = 0;
                                    }
                              }catch (Exception x) {
                                    System.out.println(x);
                                    nack++;
                              }
                        }
                  }else if(mode == 2){
                        String code = pInfo + "00";
                        while (counter < 100) {
                              txbuffer = code.getBytes();
                              p = new
DatagramPacket(txbuffer,txbuffer.length,hostAdr,sPort);
                              try {
                                    s.send(p);
                                    r.receive(q);
                                    String message = new
String(rxbuffer,0,q.getLength());
                                    System.out.println(message + "     " + code);
                                    messages.add(message);
                                    messages.add(code);
                                    ack++;
                              }catch (Exception x) {
                                    System.out.println("Packet didn't
received!");
                                    nack++;
                              }
                              code = pInfo + String.valueOf(counter);
                              counter++;
                        }
                  }
                  r.close();
                  s.close();
                  float pACK = (float)ack / (ack+nack);
                  System.out.println("Success rate: " + (float)100*pACK + "%");

//---------------------------------------ADDING_TO_FILE---------------------
-------------------------
                  FileOutputStream echoPacket = new FileOutputStream("Echo
Packets.text");
                  PrintStream prt = new PrintStream(echoPacket);
                  for (int i=0; i<responseTime.size(); i++)
                        prt.println("Packet "+(i+1)+"\t\tResponse
Time="+responseTime.get(i)+"\tSRTT="

      +srttAr.get(i)+"\tVar="+varAr.get(i)+"\t\tRTO="+rtoAr.get(i));
                  for (int i=0; i<throughputBPS.size(); i++)
                        prt.println((i+1) + ". Throughput
(BPS)="+throughputBPS.get(i));
                  for (int i=0; i<messages.size(); i+=2)
```

```
                    prt.println(messages.get(i) + "\tCode: " +
messages.get(i+1));
            prt.println("\nProbability of Occurrence: "+(float)100*pACK+"%");
            prt.close();
            System.out.println("\nFiles Created");
      }
}
```

IMAGE:

```
import java.awt.image.BufferedImage;
import java.io.ByteArrayInputStream;
import java.io.File;
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;
import javax.imageio.ImageIO;

public class image {

	ArrayList<Byte> img = new ArrayList<Byte>();
	int nack = 0, ack = 0;

	public void run(String pInfo, int sPort, int cPort, InetAddress hostAdr,
int cam) throws IOException {

		DatagramSocket s = new DatagramSocket();
		byte[] txbuffer = pInfo.getBytes();
		DatagramPacket p = new
DatagramPacket(txbuffer,txbuffer.length,hostAdr,sPort);
		s.send(p);
		DatagramSocket r = new DatagramSocket(cPort);
		r.setSoTimeout(5000);
		byte[] rxbuffer = new byte[2048];
		DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);

		System.out.println("\nLoading bytes of image...");
		for(;;) {
			try {
				r.receive(q);
				ack++;
				for(int i=0; i<q.getLength(); i++)
					img.add(rxbuffer[i]);
				if (q.getLength() != 128)
					break;
			}catch (Exception x) {
				System.out.println(x);
				nack++;
			}
		}
		r.close();
		s.close();
		float pACK = (float)ack / (ack+nack);
		System.out.println("Success rate: " + (float)100*pACK + "%");

//-----------------------------------------IMAGE_CONVERSION-------------
---------------------------------
		byte[] image = new byte[img.size()];
		for (int i=0; i<img.size(); i++)
			image[i] = img.get(i);
		BufferedImage bufferedImage = ImageIO.read(new
ByteArrayInputStream(image));
		if (cam == 1)
```

```
                    ImageIO.write(bufferedImage,"jpeg",new
File("C:\\Users\\Marios\\Java\\Networks_II\\Image_1.jpg"));
             else
                    ImageIO.write(bufferedImage,"jpeg",new
File("C:\\Users\\Marios\\Java\\Networks_II\\Image_2.jpg"));
             System.out.println("\nImage Created");
      }
}
```

AUDIO DPCM:

```
import java.io.ByteArrayInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;
import javax.sound.sampled.AudioFileFormat;
import javax.sound.sampled.AudioFormat;
import javax.sound.sampled.AudioInputStream;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.LineUnavailableException;
import javax.sound.sampled.SourceDataLine;

public class audioDPCM {

	int nack=0, ack=0, bytes, nib1, nib2, dif1, dif2, sample1, sample2=0;
	ArrayList<Integer> differences = new ArrayList<Integer>();
	ArrayList<Integer> samples = new ArrayList<Integer>();
	ArrayList<Byte> audioBuffer = new ArrayList<Byte>();

	public void run(String pInfo, int sPort, int cPort, InetAddress hostAdr,
int num) throws LineUnavailableException, IOException {

		DatagramSocket s = new DatagramSocket();
		byte[] txbuffer = pInfo.getBytes();
		DatagramPacket p = new
DatagramPacket(txbuffer,txbuffer.length,hostAdr,sPort);
		s.send(p);
		DatagramSocket r = new DatagramSocket(cPort);
		r.setSoTimeout(3000);
		byte[] rxbuffer = new byte[2048];
		DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);

//-------------------------------------PROCESS-------------------------------
--------
		System.out.println("\nLoading/Decoding bytes of audio...");
		for(int i=0; i<num; i++) {
			try {
				r.receive(q);
				ack++;
				for (int j=0; j<q.getLength(); j++) {
					bytes = rxbuffer[j];
					nib1 = (bytes & 0xF0) >> 4;
					nib2 = (bytes & 0x0F);
					dif1 = nib1-8;
					dif2 = nib2-8;
					sample1 = sample2 + (dif1*1);		//
[Quantization Step (b) = 1, MUST BE =>1]
					sample2 = sample1 + (dif2*1);
					audioBuffer.add((byte)sample1);
					audioBuffer.add((byte)sample2);
					differences.add(dif1);
					differences.add(dif2);
					samples.add(sample1);
```

```
                                samples.add(sample2);
                        }
                }catch (Exception x) {
                        System.out.println(x);
                        nack++;
                }
        }
        s.close();
        r.close();
        float pACK =(float)ack / (ack+nack);
        System.out.println("...Process Finished!");
        System.out.println("Success rate: " + (float)100*pACK + "%");


//---------------------------------CONVERT_TO_BYTE_ARRAY----------------------
----
        byte[] audio = new byte[audioBuffer.size()];
        for(int k=0; k<audioBuffer.size(); k++)
                audio[k] = audioBuffer.get(k);

//-----------------------------------PLAYING_AUDIO----------------------------
-----
        System.out.println("Playing audio...");
        AudioFormat lnrPCM = new AudioFormat(8000,8,1,true,false);
        SourceDataLine lnrOut = AudioSystem.getSourceDataLine(lnrPCM);
        lnrOut.open(lnrPCM, 32000);
        lnrOut.start();
        lnrOut.write(audio,0,audio.length);
        lnrOut.stop();
        lnrOut.close();
        System.out.println("...Player stopped!");

//---------------------------------------ADDING_TO_FILE---------------------
-------------------------
        FileOutputStream audioFile = new FileOutputStream("Audio
(DPCM).wav");
        FileOutputStream audioDPCM = new FileOutputStream("Audio
(DPCM).text");
        PrintStream prt = new PrintStream(audioDPCM);
        prt.println("Audio - DPCM");
        for (int i=0; i<samples.size(); i++)

     prt.println("Sample"+(i+1)+"="+samples.get(i)+"\tDifference"+(i+1)+"="+di
fferences.get(i));
        prt.println("\nProbability of Occurrence: "+(float)100*pACK+"%");
        prt.close();
        ByteArrayInputStream inStream = new ByteArrayInputStream(audio);
        AudioInputStream audioInStream = new AudioInputStream(inStream,
lnrPCM, audio.length);
        AudioSystem.write(audioInStream, AudioFileFormat.Type.WAVE,
audioFile);
        System.out.println("\nFiles Created");
     }
}
```

AUDIO AQDPCM:

```
import java.io.ByteArrayInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;
import javax.sound.sampled.AudioFileFormat;
import javax.sound.sampled.AudioFormat;
import javax.sound.sampled.AudioInputStream;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.LineUnavailableException;
import javax.sound.sampled.SourceDataLine;

public class audioAQDPCM {

	FileOutputStream audioFile, audioAQDPCM;
	int nack=0, ack=0, m, b, bytes, nib1, nib2, dif1, dif2, sample1, sample2;
	ArrayList<Integer> ms = new ArrayList<Integer>();
	ArrayList<Integer> betas = new ArrayList<Integer>();
	ArrayList<Integer> differences = new ArrayList<Integer>();
	ArrayList<Integer> samples = new ArrayList<Integer>();
	ArrayList<Byte> audioBuffer = new ArrayList<Byte>();

	public void run(String pInfo, int sPort, int cPort, InetAddress hostAdr,
int num) throws LineUnavailableException, IOException {

		DatagramSocket s = new DatagramSocket();
		byte[] txbuffer = pInfo.getBytes();
		DatagramPacket p = new
DatagramPacket(txbuffer,txbuffer.length,hostAdr,sPort);
		s.send(p);
		DatagramSocket r = new DatagramSocket(cPort);
		r.setSoTimeout(3000);
		byte[] rxbuffer = new byte[2048];
		DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);


//-------------------------------------PROCESS-------------------------------
--------
		System.out.println("\nLoading/Decoding bytes of audio...");
		for(int i=0; i<num; i++) {
			try {
				r.receive(q);
				ack++;
				m = (rxbuffer[1] << 8 | (rxbuffer[0] & 0xFF));
				b = (rxbuffer[3] << 8 | (rxbuffer[2] & 0xFF));
				ms.add(m);
				betas.add(b);
				for (int j=4; j<q.getLength(); j++) {
					bytes = rxbuffer[j];
					nib1 = (bytes & 0xF0) >> 4;
					nib2 = (bytes & 0x0F);
					dif1 = nib1-8;
					dif2 = nib2-8;
```

```
                                    sample1 = (dif1*b) + m;     // [Quantization
Step (b)]
                                    sample2 = (dif2*b) + m;
                                    audioBuffer.add((byte)(sample1 & 0xFF));
                                    audioBuffer.add((byte)(sample1 >> 8));
                                    audioBuffer.add((byte)(sample2 & 0xFF));
                                    audioBuffer.add((byte)(sample2 >> 8));
                                    differences.add(dif1);
                                    differences.add(dif2);
                                    samples.add(sample1);
                                    samples.add(sample2);
                             }
                      }catch (Exception x) {
                             System.out.println(x);
                             nack++;
                      }
              }
              s.close();
              r.close();
              float pACK = (float)ack / (ack+nack);
              System.out.println("...Process Finished!");
              System.out.println("Success rate: " + (float)100*pACK + "%");


//---------------------------------CONVERT_TO_BYTE_ARRAY----------------------
----
              byte[] audio = new byte[audioBuffer.size()];
              for(int k=0; k<audioBuffer.size(); k++)
                     audio[k] = audioBuffer.get(k);

//-----------------------------------PLAYING_AUDIO----------------------------
-----
              System.out.println("Playing audio...");
              AudioFormat lnrPCM = new AudioFormat(8000,16,1,true,false);
              SourceDataLine lnrOut = AudioSystem.getSourceDataLine(lnrPCM);
              lnrOut.open(lnrPCM, audio.length);
              lnrOut.start();
              lnrOut.write(audio,0,audio.length);
              lnrOut.stop();
              lnrOut.close();
              System.out.println("...Player stopped!");

//---------------------------------------ADDING_TO_FILE---------------------
-------------------------
              audioFile = new FileOutputStream("Audio (AQDPCM).wav");
              audioAQDPCM = new FileOutputStream("Audio (AQDPCM).text");
              PrintStream prt = new PrintStream(audioAQDPCM);
              prt.println("Audio - AQDPCM");
              for (int i=0; i<samples.size(); i++)
                     prt.println("Sample"+(i+1)
+"="+samples.get(i)+"\tDifference"+(i+1)+"="+differences.get(i));
              for (int i=0; i<ms.size(); i++)
                     prt.println("\tMean"+(i+1)
+"="+ms.get(i)+"\tBeta"+(i+1)+"="+betas.get(i));
              prt.println("\nProbability of Occurrence: "+(float)100*pACK+"%");
              prt.close();
              ByteArrayInputStream inStream = new ByteArrayInputStream(audio);
```

```
            AudioInputStream audioInStream = new AudioInputStream(inStream,
lnrPCM, audio.length);
            AudioSystem.write(audioInStream, AudioFileFormat.Type.WAVE,
audioFile);
            System.out.println("\nFiles Created!");
      }
}
```

ITHAKI COPTER TCP:

```
import java.io.BufferedReader;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.net.InetAddress;
import java.net.Socket;
import java.util.ArrayList;

public class ithakiCopter_I {

	ArrayList<String> messages= new ArrayList<String>();
	long startTime = System.currentTimeMillis();

	public void run(int portNum, InetAddress hostAdr, String request) throws
IOException {

		Socket s = new Socket(hostAdr, portNum);

		PrintStream out = new PrintStream(s.getOutputStream());
		BufferedReader in = new BufferedReader(new
InputStreamReader(s.getInputStream()));

		out.println(request);
		while(System.currentTimeMillis() < startTime+60100) {
			in.skip(427);//-------------------------length of the
message
			try {
				String line = in.readLine();
				while (line != null) {
					System.out.println(line);
					messages.add(line);
					line = in.readLine();
				}
				run(portNum, hostAdr, "AUTO FLIGHTLEVEL=000
LMOTOR=000 RMOTOR=000 PILOT \r\n");

			}catch(Exception x) {
				System.out.println(x);
			}
		}
		s.close();
		out.close();
		in.close();

//---------------------------------------ADDING_TO_FILE---------------------
-------------------------
		FileOutputStream ithakiCopterFile = new FileOutputStream("Ithaki
Copter.text");
		PrintStream prt = new PrintStream(ithakiCopterFile);
		prt.println("Ithaki Copter");
		for (int i=0; i<messages.size(); i++)
			prt.println(messages.get(i));
		prt.close();
	}
}
```

ITHAKI COPTER UDP:

```
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;

public class ithakiCopter_II {
	ArrayList<String> messages= new ArrayList<String>();
	int ack=0, nack=0;
	long strTime = System.currentTimeMillis();

	public void run(String pInfo, int sPort, int cPort, InetAddress hostAdr)
throws IOException {

		DatagramSocket s = new DatagramSocket();
		byte[] txbuffer = pInfo.getBytes();
		DatagramPacket p = new
DatagramPacket(txbuffer,txbuffer.length,hostAdr,sPort);
		DatagramSocket r = new DatagramSocket(cPort);
		r.setSoTimeout(3000);
		byte[] rxbuffer = new byte[2048];
		DatagramPacket q = new DatagramPacket(rxbuffer,rxbuffer.length);

		while (System.currentTimeMillis() < (strTime+65000)) {
			try {
				s.send(p);
				r.receive(q);
				String message = new
String(rxbuffer,0,q.getLength());
				System.out.println(message);
				messages.add(message);
				ack++;
			}catch (Exception x) {
				System.out.println(x);
				nack++;
			}
		}
		System.out.println("Procces Finished!");
		float pACK = (float)ack / (ack+nack);
		s.close();
		r.close();
//---------------------------------------ADDING_TO_FILE---------------------
-------------------------
		FileOutputStream ithakiCopterFile = new FileOutputStream("Ithaki
Copter (UDP).text");
		PrintStream prt = new PrintStream(ithakiCopterFile);
		prt.println("Ithaki Copter");
		for (int i=0; i<messages.size(); i++)
			prt.println(messages.get(i));
		prt.println("\nProbability of Occurrence: "+(float)100*pACK+"%");
		prt.close();
		System.out.println("\nFiles Created");
	}
}
```

OBD II – CAR FAULT DIAGNOSTICS:

```
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.PrintStream;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;
import java.util.Scanner;

public class carFaultDiag_II {

    int ack=0, nack=0;

    public void run(String pInfo, int sPort, int cPort, InetAddress hostAdr)
throws IOException {

            Scanner scn = new Scanner(System.in);
            System.out.print("\nChoose request for OBD-II:\n1. Engine Run
Time\n"
                        + "2. Intake Air Temperature\n3. Throttle Position
\n"
                        + "4. Engine RPM\n5. Vehicle Speed\n6. Coolant
Temperature\n"
                        + "ENTER 'exit' to stop.  \n\nENTER REQUEST'S NAME:
");
            String request = scn.nextLine();

            for(;;) {
                  float formula=0;
                  String code = "";
                  if (request.equals("Engine Run Time"))
                        code = pInfo + "OBD=01 1F";
                  else if (request.equals("Intake Air Temperature"))
                        code = pInfo + "OBD=01 0F";
                  else if (request.equals("Throttle Position"))
                        code = pInfo + "OBD=01 11";
                  else if (request.equals("Engine RPM"))
                        code = pInfo + "OBD=01 0C";
                  else if (request.equals("Vehicle Speed"))
                        code = pInfo + "OBD=01 0D";
                  else if (request.equals("Coolant Temperature"))
                        code = pInfo + "OBD=01 05";
                  else if (request.equals("exit"))
                              break;

                  FileOutputStream file = new FileOutputStream("OBD_II
"+request+".text");
                  ArrayList<Float> formulaValue = new ArrayList<Float>();

                  DatagramSocket s = new DatagramSocket();
                  byte[] txbuffer = code.getBytes();
                  DatagramPacket p = new
DatagramPacket(txbuffer,txbuffer.length,hostAdr,sPort);
                  DatagramSocket r = new DatagramSocket(cPort);
                  r.setSoTimeout(3000);
```

```
                byte[] rxbuffer = new byte[5000];
                DatagramPacket q = new
DatagramPacket(rxbuffer,rxbuffer.length);

                long startTime = System.currentTimeMillis();
                while (System.currentTimeMillis() < startTime+40000) {
                        try {
                                s.send(p);
                                r.receive(q);
                                ack++;
                                String message = new
String(rxbuffer,0,q.getLength());
                                System.out.println(message);
                                if ((request.equals("Engine Run Time")) ||
(request.equals("Engine RPM"))) {
                                        String XX = "", YY = "";
                                        XX += (char)rxbuffer[6];
                                        XX += (char)rxbuffer[7];
                                        YY += (char)rxbuffer[9];
                                        YY += (char)rxbuffer[10];
                                        if (request.equals("Engine Run Time"))
                                                formula =
(float)256*Integer.parseInt(XX,16)+Integer.parseInt(YY,16);
                                        else
                                                formula =
(float)(((Integer.parseInt(XX,16)*256)+Integer.parseInt(YY,16))/4);
                                }else {
                                        String XX = "";
                                        XX += (char)rxbuffer[6];
                                        XX += (char)rxbuffer[7];
                                        if (request.equals("Intake Air
Temperature") || (request.equals("Coolant Temperature")))
                                                formula =
(float)Integer.parseInt(XX,16) - 40;
                                        else if (request.equals("Throttle
Position"))
                                                formula =
(float)Integer.parseInt(XX,16)*100/255;
                                        else if (request.equals("Vehicle
Speed"))
                                                formula =
(float)Integer.parseInt(XX,16);
                                }
                                formulaValue.add(formula);

                        }catch (Exception x) {
                                System.out.println(x);
                                nack++;
                        }
                }
                r.close();
                s.close();
                float pACK = (float)ack / (ack+nack);
                System.out.println("Success rate: " + (float)100*pACK +
"%");

//---------------------------------------ADDING_TO_FILE-------------------
-------------------------
```

```
                    PrintStream prt = new PrintStream(file);
                    prt.println(request);
                    for (int i=0; i<formulaValue.size(); i++)
                           prt.println((i+1) + ". " + formulaValue.get(i));
                    prt.println("\nProbability of Occurrence:
"+(float)100*pACK+"%");
                    prt.close();

                    System.out.println("\nFile Created");
                    System.out.print("\nChoose next request for OBD-II:\n1.
Engine Run Time\n"
                                  + "2. Intake Air Temperature\n3. Throttle
Position \n"
                                  + "4. Engine RPM\n5. Vehicle Speed\n6.
Coolant temperature \n\nENTER REQUEST'S NAME: ");
                    request = scn.nextLine();
             }
             scn.close();
             System.out.println("Procces Finished!");
      }
}
```