CSI THOOTHUKUDI-NAZARETH DIOCESE

DR.G.U.POPE COLLEGE OF ENGINEERING

POPE NAGAR SAWYERPURAM-628 251

Register No	:

Certified that this is the bonafide record of work done by

Selvan/Selvi ………………………………………………………………………… of ……….

Semester ……….. branch for the lab ……………………………………………………

During the year…………………

Staff In-charge **H. O.D**

Submitted for the university practical Examination held on …………..

Internal Examiner **External Examiner**

S.NO	DATE	TITLE	PAGE	MARK	SIGN
1			3		
2			5		
3			7		
4			13		
5			16		
6			19		
7			22		
8			25		
9			28		
10			31		
11			34		

EX NO: 1	MAKE STUDY USING COMMANDS LIKE TCPDUMP,NETSTAT,IFCONFIG,NSLOOKUP,TRACEROUTE
DATE:	

Aim : To use commands like tcpdump,netstat,ifconfig,nslookup,traceroute,using a network protocol analyzer.

Algorithm :

1. Use the linux for the commands fast executon.
2. Open terminal & switch to root user.
3. Install the required tools to work.
4. Use traffic analyzer commands & observe the executions.

Commands :

```
sudo -i
apt install net-tools
ifconfig
tcpdump -d
tcpdump -i
netstat -a
netstat -l
netstat -s
netstat -i
netstat -r
nslookup 8.8.8.8
nslookup google.com 8.8.8.8
apt install inetutils-traceroute
traceroute google.com
apt install wireshark
wireshark
```

Output :

Result:

Thus to use commands like tcpdump,netstat,ifconfig,nslookup,traceroute,using a network protocol analyzer have been analyzed successfully.

EX NO: 2	WRITE AN HTTPS WEB CLIENT PROGRAM TO DOWNLOAD A WEB PAGE USING TCP SOCKETS.
DATE:	

Aim : To write an HTTPS web client program to download a web page using TCP sockets.

Algorithm:

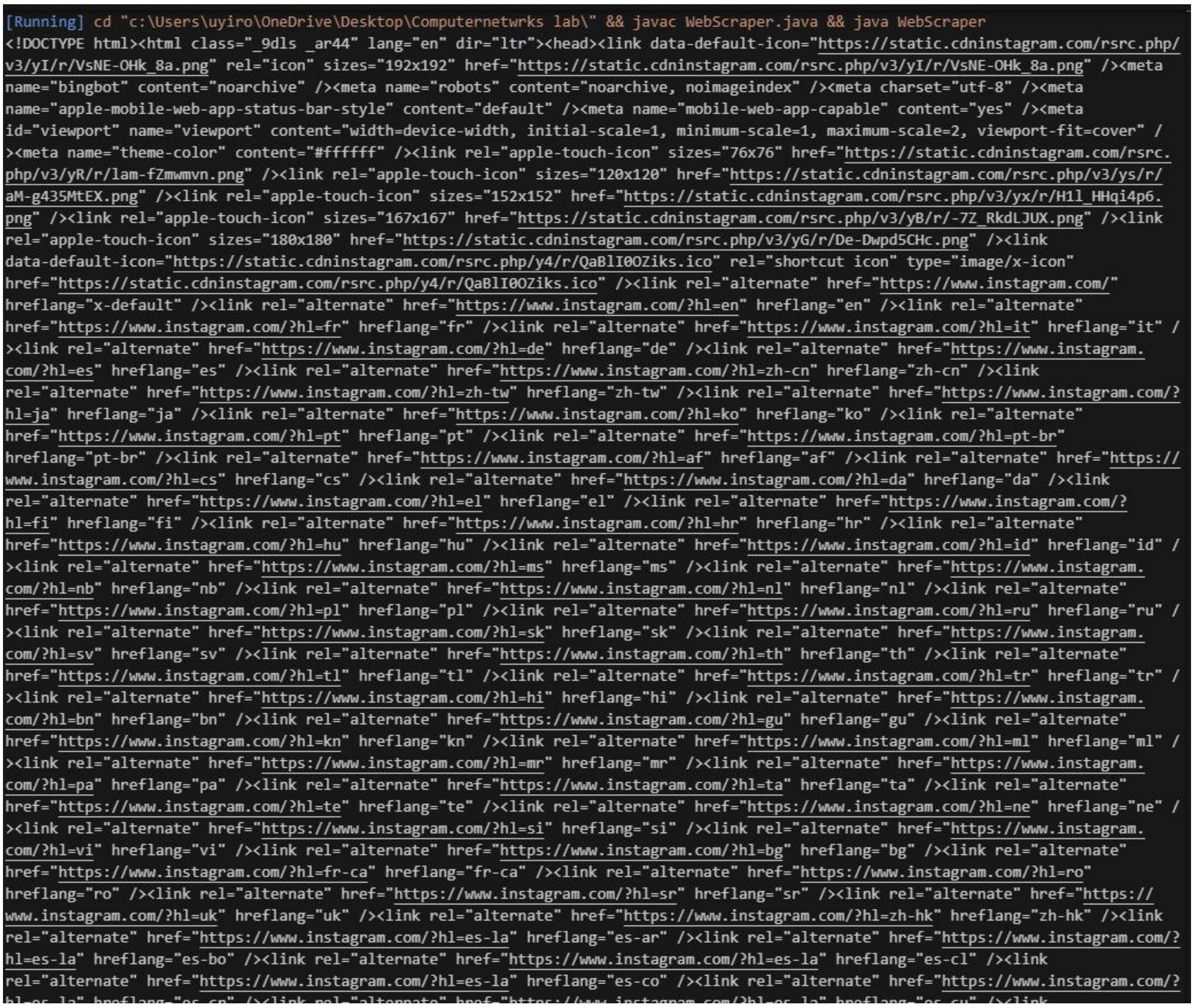
1. Import the necessary packages.
2. Initialize URI & URL.
3. Create a bufferreader object to read the URL stream.
4. Create a bufferreader object to write data into it.
5. Console the program & data lines
6. Observe the executed output.

Program :

```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.InputStreamReader;
import java.net.URI;
import java.net.URL;

public class WebScraper {
   public static void main(String[] args) throws Exception {

      URI uri = new URI("https://www.instagram.com");
      URL url = uri.toURL();


      BufferedReader reader = new BufferedReader(new InputStreamReader(url.openStream()));


      BufferedWriter writer = new BufferedWriter(new FileWriter("data.html"));

      String line;

      while ((line = reader.readLine()) != null) {
         System.out.println(line);
         writer.write(line);
         writer.newLine();
      }


      reader.close();
      writer.close();
   }
```

}

Output :



```
[Running] cd "c:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\" && javac WebScraper.java && java WebScraper
<!DOCTYPE html><html class="_9dls _ar44" lang="en" dir="ltr"><head><link data-default-icon="https://static.cdninstagram.com/rsrc.php/
v3/yI/r/VsNE-OHk_8a.png" rel="icon" sizes="192x192" href="https://static.cdninstagram.com/rsrc.php/v3/yI/r/VsNE-OHk_8a.png" /><meta
name="bingbot" content="noarchive" /><meta name="robots" content="noarchive, noimageindex" /><meta charset="utf-8" /><meta
name="apple-mobile-web-app-status-bar-style" content="default" /><meta name="mobile-web-app-capable" content="yes" /><meta
id="viewport" name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1, maximum-scale=2, viewport-fit=cover" /
><meta name="theme-color" content="#ffffff" /><link rel="apple-touch-icon" sizes="76x76" href="https://static.cdninstagram.com/rsrc.
php/v3/yR/r/lam-fZmwmvn.png" /><link rel="apple-touch-icon" sizes="120x120" href="https://static.cdninstagram.com/rsrc.php/v3/ys/r/
aM-g435MtEX.png" /><link rel="apple-touch-icon" sizes="152x152" href="https://static.cdninstagram.com/rsrc.php/v3/yx/r/H1l_HHqi4p6.
png" /><link rel="apple-touch-icon" sizes="167x167" href="https://static.cdninstagram.com/rsrc.php/v3/yB/r/-7Z_RkdLJUX.png" /><link
rel="apple-touch-icon" sizes="180x180" href="https://static.cdninstagram.com/rsrc.php/v3/yG/r/De-Dwpd5CHc.png" /><link
data-default-icon="https://static.cdninstagram.com/rsrc.php/y4/r/QaBlI0OZiks.ico" rel="shortcut icon" type="image/x-icon"
href="https://static.cdninstagram.com/rsrc.php/y4/r/QaBlI0OZiks.ico" /><link rel="alternate" href="https://www.instagram.com/"
hreflang="x-default" /><link rel="alternate" href="https://www.instagram.com/?hl=en" hreflang="en" /><link rel="alternate"
href="https://www.instagram.com/?hl=fr" hreflang="fr" /><link rel="alternate" href="https://www.instagram.com/?hl=it" hreflang="it" /
><link rel="alternate" href="https://www.instagram.com/?hl=de" hreflang="de" /><link rel="alternate" href="https://www.instagram.
com/?hl=es" hreflang="es" /><link rel="alternate" href="https://www.instagram.com/?hl=zh-cn" hreflang="zh-cn" /><link
rel="alternate" href="https://www.instagram.com/?hl=zh-tw" hreflang="zh-tw" /><link rel="alternate" href="https://www.instagram.com/?
hl=ja" hreflang="ja" /><link rel="alternate" href="https://www.instagram.com/?hl=ko" hreflang="ko" /><link rel="alternate"
href="https://www.instagram.com/?hl=pt" hreflang="pt" /><link rel="alternate" href="https://www.instagram.com/?hl=pt-br"
hreflang="pt-br" /><link rel="alternate" href="https://www.instagram.com/?hl=af" hreflang="af" /><link rel="alternate" href="https://
www.instagram.com/?hl=cs" hreflang="cs" /><link rel="alternate" href="https://www.instagram.com/?hl=da" hreflang="da" /><link
rel="alternate" href="https://www.instagram.com/?hl=el" hreflang="el" /><link rel="alternate" href="https://www.instagram.com/?
hl=fi" hreflang="fi" /><link rel="alternate" href="https://www.instagram.com/?hl=hr" hreflang="hr" /><link rel="alternate"
href="https://www.instagram.com/?hl=hu" hreflang="hu" /><link rel="alternate" href="https://www.instagram.com/?hl=id" hreflang="id" /
><link rel="alternate" href="https://www.instagram.com/?hl=ms" hreflang="ms" /><link rel="alternate" href="https://www.instagram.
com/?hl=nb" hreflang="nb" /><link rel="alternate" href="https://www.instagram.com/?hl=nl" hreflang="nl" /><link rel="alternate"
href="https://www.instagram.com/?hl=pl" hreflang="pl" /><link rel="alternate" href="https://www.instagram.com/?hl=ru" hreflang="ru" /
><link rel="alternate" href="https://www.instagram.com/?hl=sk" hreflang="sk" /><link rel="alternate" href="https://www.instagram.
com/?hl=sv" hreflang="sv" /><link rel="alternate" href="https://www.instagram.com/?hl=th" hreflang="th" /><link rel="alternate"
href="https://www.instagram.com/?hl=tl" hreflang="tl" /><link rel="alternate" href="https://www.instagram.com/?hl=tr" hreflang="tr" /
><link rel="alternate" href="https://www.instagram.com/?hl=hi" hreflang="hi" /><link rel="alternate" href="https://www.instagram.
com/?hl=bn" hreflang="bn" /><link rel="alternate" href="https://www.instagram.com/?hl=gu" hreflang="gu" /><link rel="alternate"
href="https://www.instagram.com/?hl=kn" hreflang="kn" /><link rel="alternate" href="https://www.instagram.com/?hl=ml" hreflang="ml" /
><link rel="alternate" href="https://www.instagram.com/?hl=mr" hreflang="mr" /><link rel="alternate" href="https://www.instagram.
com/?hl=pa" hreflang="pa" /><link rel="alternate" href="https://www.instagram.com/?hl=ta" hreflang="ta" /><link rel="alternate"
href="https://www.instagram.com/?hl=te" hreflang="te" /><link rel="alternate" href="https://www.instagram.com/?hl=ne" hreflang="ne" /
><link rel="alternate" href="https://www.instagram.com/?hl=si" hreflang="si" /><link rel="alternate" href="https://www.instagram.
com/?hl=vi" hreflang="vi" /><link rel="alternate" href="https://www.instagram.com/?hl=bg" hreflang="bg" /><link rel="alternate"
href="https://www.instagram.com/?hl=fr-ca" hreflang="fr-ca" /><link rel="alternate" href="https://www.instagram.com/?hl=ro"
hreflang="ro" /><link rel="alternate" href="https://www.instagram.com/?hl=sr" hreflang="sr" /><link rel="alternate" href="https://
www.instagram.com/?hl=uk" hreflang="uk" /><link rel="alternate" href="https://www.instagram.com/?hl=zh-hk" hreflang="zh-hk" /><link
rel="alternate" href="https://www.instagram.com/?hl=es-la" hreflang="es-ar" /><link rel="alternate" href="https://www.instagram.com/?
hl=es-la" hreflang="es-bo" /><link rel="alternate" href="https://www.instagram.com/?hl=es-la" hreflang="es-cl" /><link
rel="alternate" href="https://www.instagram.com/?hl=es-la" hreflang="es-co" /><link rel="alternate" href="https://www.instagram.com/?
hl=es-la" hreflang="es-cr" /><link rel="alternate" href="https://www.instagram.com/?hl=es-la" hreflang="es-cu" /><link
```

Result : Thus to write an https web client program to download a web page using tcp sockets has been executed successfully.

EX NO: 3	CREATE & RUN ECHOSERVER, ECHOCLIENT, FILETRANSFERSERVER, FILETRANSFERCLIENT USING TCP SOCKETS.
DATE:	

Aim : To create & run echoserver,echoclient,filetransferserver,filetransferclient using TCP sockets.

.

Algorithm[ES]:
1. Import the necessary packages.
2. Set the port numbers.
3. Initialize the server sockets.
4. Enter sn infinite loop to hsndle incoming clients.
5. Setup inut & output streams.
6. Process client messages.

Algorithm[EC]:
1. Import the necessary packages.
2. Set the host and port.
3. Establish the connection to server.
4. Setup I/O stream.
5. Display connection success messages.
6. Read the user inpt.
7. Close the connection when done.

Algorithm[FTS]:
1. Import the necessary packages.
2. Set the port number.
3. Initialize the server socket.
4. Setup I/O file stream.
5. Transfer data from client to server.
6. Close resources after transaction.

Algorithm[FTC]:
1. import the necessary packages.
2. Set the host, port, filepath.
3. Establish a connection to server.
4. Setup I/O stream.
5. Transfer the file data.
6. Close resources after transaction.
7. Display success message.

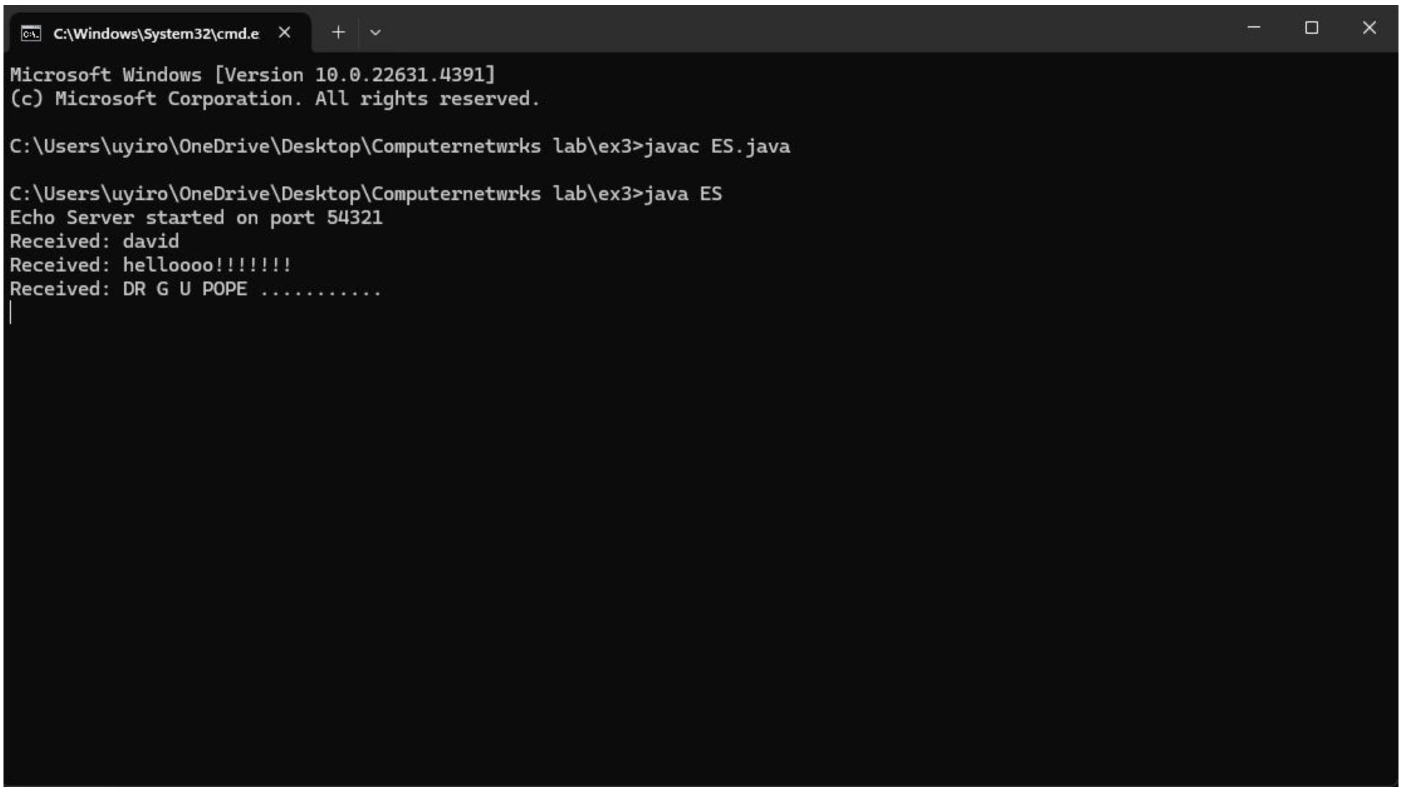
BioData :

Name : DavidJones
Father’s Name : Nicholas
Ph No : 9488209176
Mail ID : jesustheimmaculatta@gmail.com
DOB : 20.06.2005
Gender : Male
Marital Status : Unmarried
Qualification : Diploma in Computer Applications
Experience : 0 yrs
Address :5B/403, Coats Nagar, Vallinayagapuram, tuty

Program :
a) EchoServer

```
import java.io.*;
import java.net.*;

public class EchoServer {
    public static void main(String[] args) {
        int port = _ _ _ _ _;
        try (ServerSocket serverSocket = new ServerSocket(port)) {
            System.out.println("Echo Server started on port " + port);

            while (true) {
                try (Socket clientSocket = serverSocket.accept();
                     BufferedReader in = new BufferedReader(new
InputStreamReader(clientSocket.getInputStream()));
                     PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true)) {

                    String message;
                    while ((message = in.readLine()) != null) {
                        System.out.println("Received: " + message);
                        out.println("Echo: " + message);
                    }
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output :

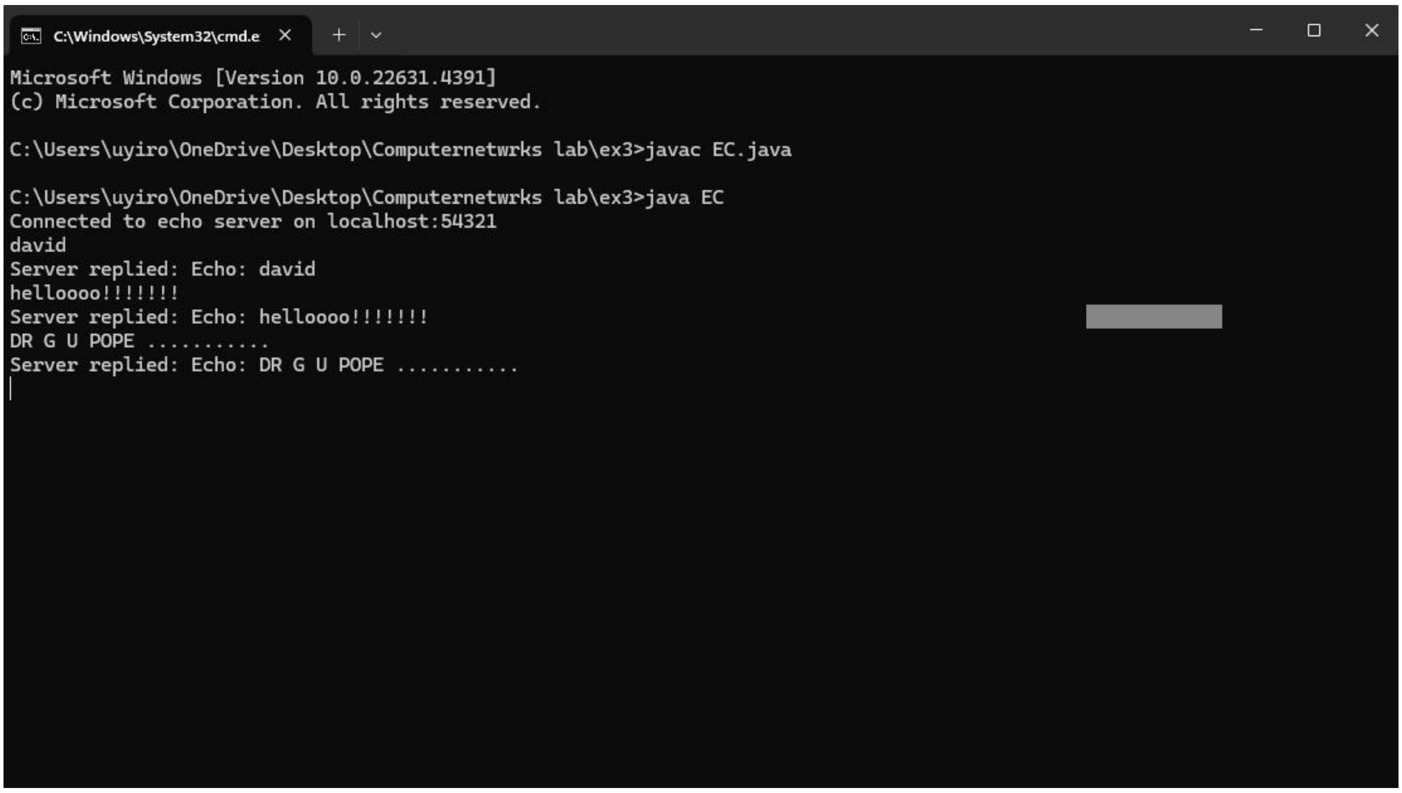


```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>javac ES.java

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>java ES
Echo Server started on port 54321
Received: david
Received: helloooo!!!!!!!
Received: DR G U POPE ..........
```

b) EchoClient

```
import java.io.*;
import java.net.*;

public class EchoClient {
    public static void main(String[] args) {
        String host = "localhost";
        int port = _ _ _ _ _;

        try (Socket socket = new Socket(host, port);
             BufferedReader in = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
             PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
             BufferedReader consoleIn = new BufferedReader(new
InputStreamReader(System.in))) {

            System.out.println("Connected to echo server on " + host + ":" + port);

            String userInput;
            while ((userInput = consoleIn.readLine()) != null) {
                out.println(userInput);
                System.out.println("Server replied: " + in.readLine());
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output :



```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>javac EC.java

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>java EC
Connected to echo server on localhost:54321
david
Server replied: Echo: david
helloooo!!!!!!!
Server replied: Echo: helloooo!!!!!!!
DR G U POPE ...........
Server replied: Echo: DR G U POPE ..........
```

c) FileTransferServer

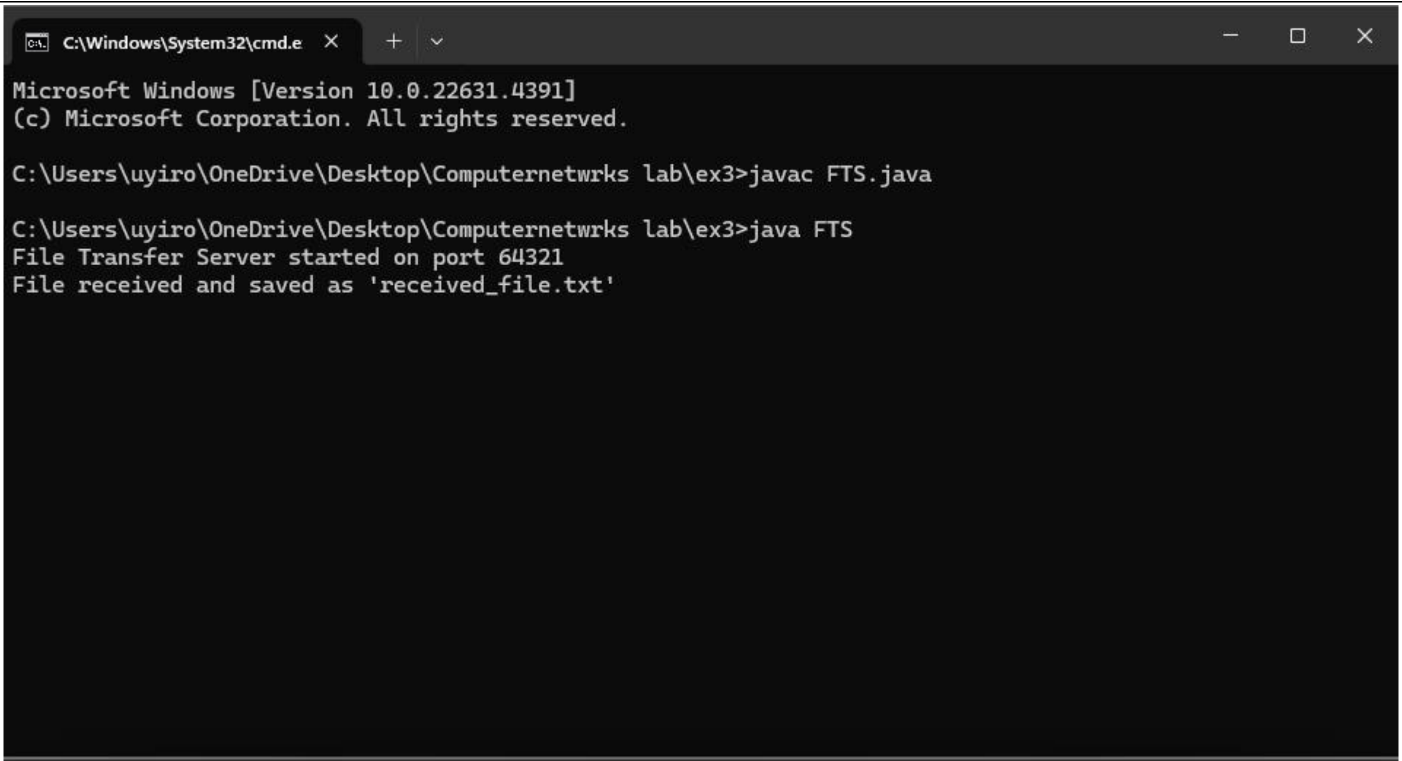
```
import java.io.*;
import java.net.*;

public class FileTransferServer {
    public static void main(String[] args) {
        int port = _ _ _ _ _;
        try (ServerSocket serverSocket = new ServerSocket(port)) {
            System.out.println("File Transfer Server started on port " + port);

            while (true) {
                try (Socket clientSocket = serverSocket.accept();
                     InputStream in = clientSocket.getInputStream();
                     FileOutputStream fileOut = new FileOutputStream("received_        ")) {

                    byte[] buffer = new byte[4096];
                    int bytesRead;
                    while ((bytesRead = in.read(buffer)) != -1) {
```

```
                    fileOut.write(buffer, 0, bytesRead);
                }
                System.out.println("File received and saved as 'received_file.txt'");
            }
        }
    } catch (IOException e) {
        e.printStackTrace();
    }
  }
}
```

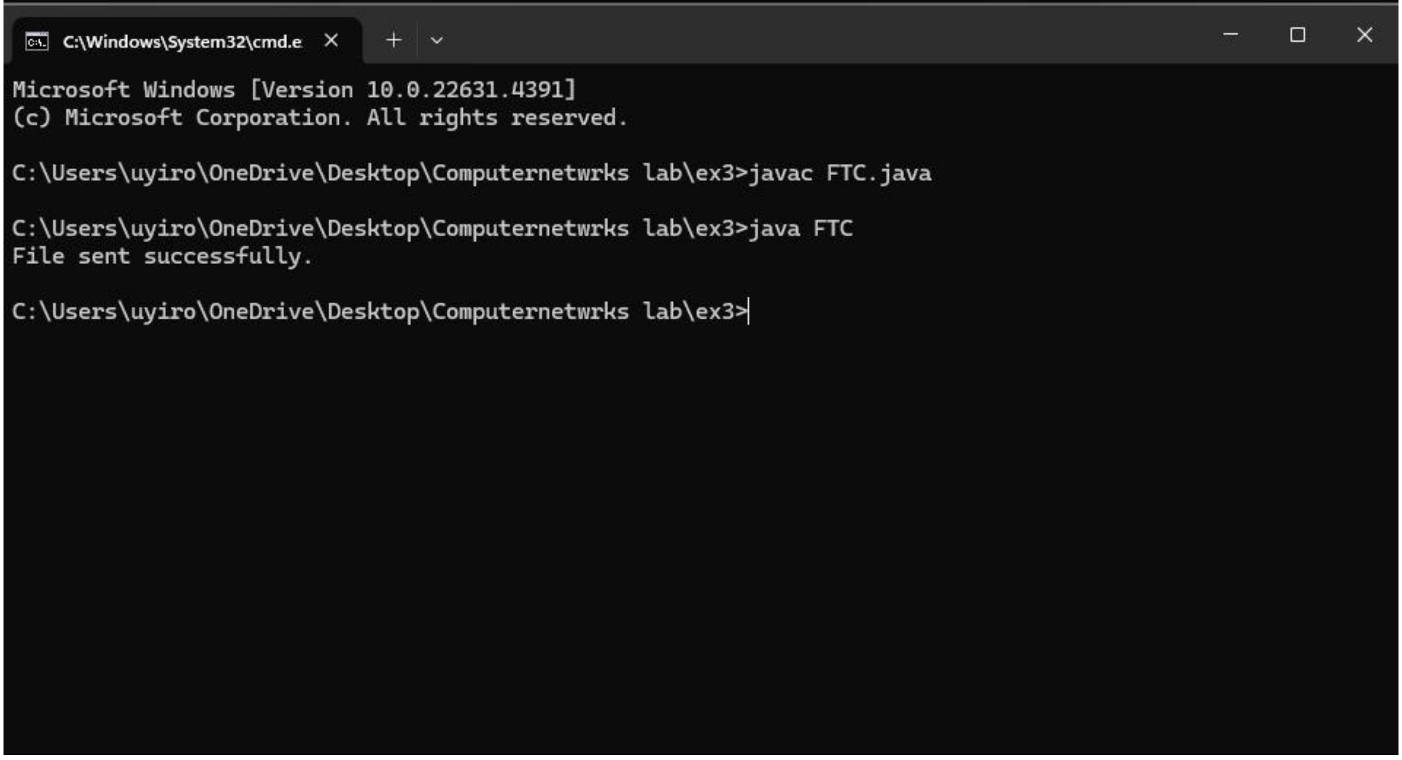
d) FileTransferClient

```
import java.io.*;
import java.net.*;

public class FileTransferClient {
    public static void main(String[] args) {
        String host = "localhost";
        int port = _ _ _ _ _;
        String filePath = "                                   "; // Update with correct path

        try (Socket socket = new Socket(host, port);
             FileInputStream fileIn = new FileInputStream(filePath);
             OutputStream out = socket.getOutputStream()) {

            byte[] buffer = new byte[4096];
            int bytesRead;
            while ((bytesRead = fileIn.read(buffer)) != -1) {
                out.write(buffer, 0, bytesRead);
            }
            System.out.println("File sent successfully.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Output :



```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>javac FTS.java

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>java FTS
File Transfer Server started on port 64321
File received and saved as 'received_file.txt'
```



```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>javac FTC.java

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>java FTC
File sent successfully.

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\ex3>
```

Result : Thus to create & run echoserver,echoclient,filetransferserver,filetransferclient using TCP sockets has been recorded successfully.

EX NO: 4	WRITE CODES TO STUDY THE BASIC SIMULATION OF DNS USING UDP SOCKETS.
DATE:	

Aim: To write codes to study the basic simulation of DNS using UDP sockets.

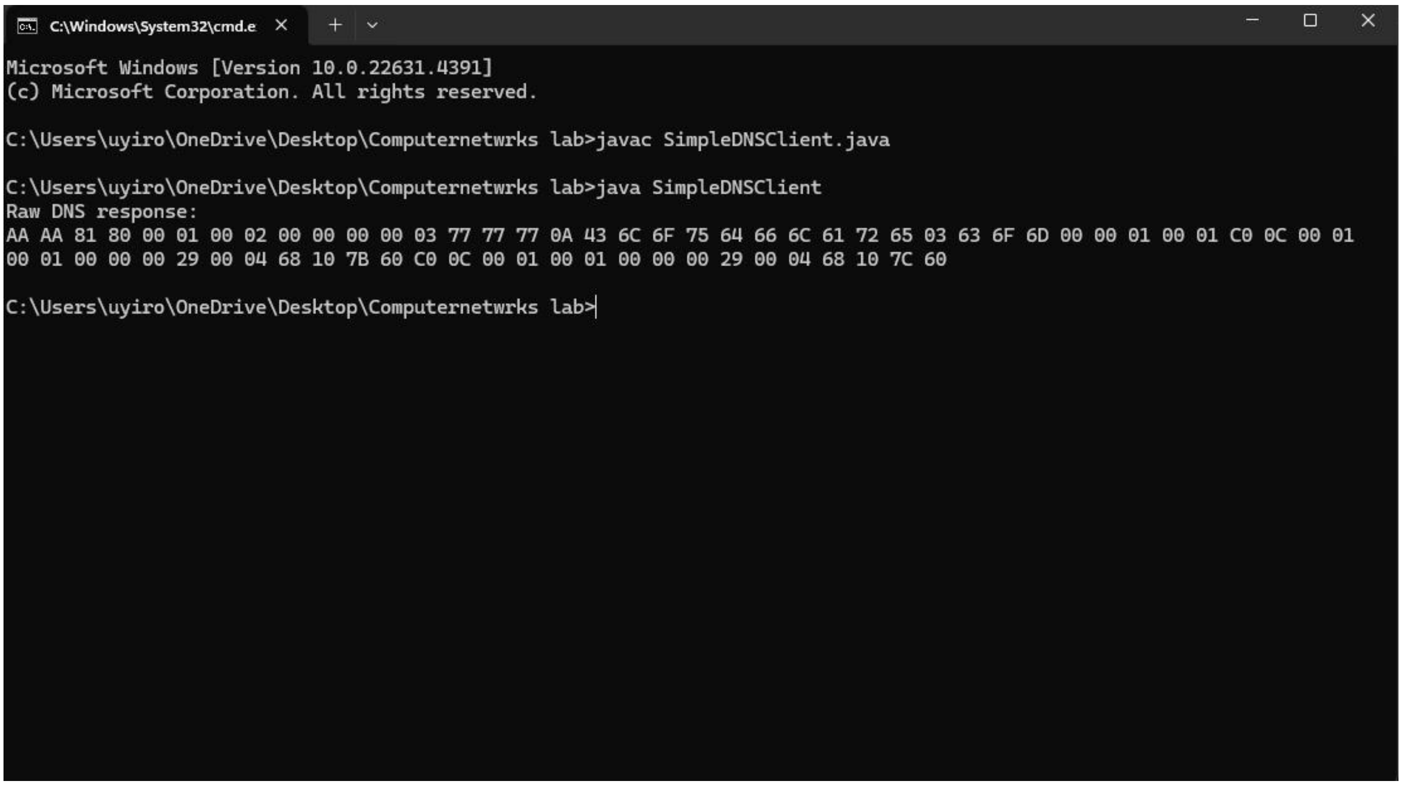
Algorithm:

1. Import the necessary packages.
2. Assign the doamin address.
3. Ensure your connection with server.
4. Request for datapackets and record the responses.
5. Catchout the occurred exceptions and relay it.
6. Note down the output for further verification.

Program:

```
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;

public class SimpleDNSClient {
    public static void main(String[] args) {
        try {
            String domain = "www.Cloudflare.com";
            InetAddress dnsServer = InetAddress.getByName("1.1.1.1");

            byte[] dnsQuery = buildDnsQuery(domain);

            DatagramSocket socket = new DatagramSocket();
            DatagramPacket requestPacket = new DatagramPacket(dnsQuery, dnsQuery.length,
dnsServer, 53);
            socket.send(requestPacket);

            byte[] buffer = new byte[512];
            DatagramPacket responsePacket = new DatagramPacket(buffer, buffer.length);
            socket.receive(responsePacket);

            socket.close();



            System.out.println("Raw DNS response: ");
            for (int i = 0; i < responsePacket.getLength(); i++) {
                System.out.print(String.format("%02X ", buffer[i]));
            }
            System.out.println();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static byte[] buildDnsQuery(String domain) throws Exception {
        byte[] header = {
            (byte) 0xAA, (byte) 0xAA,
            (byte) 0x01, (byte) 0x00,
            (byte) 0x00, (byte) 0x01,
            (byte) 0x00, (byte) 0x00,
            (byte) 0x00, (byte) 0x00,
            (byte) 0x00, (byte) 0x00
        };

        byte[] question = new byte[domain.length() + 2 + 4];
        String[] labels = domain.split("\\.");
        int pos = 0;
        for (String label : labels) {
            question[pos++] = (byte) label.length();
            for (char c : label.toCharArray()) {
                question[pos++] = (byte) c;
            }
        }

        question[pos++] = 0x00;
        question[pos++] = 0x00; question[pos++] = 0x01;
        question[pos++] = 0x00; question[pos++] = 0x01;

        byte[] dnsRequest = new byte[header.length + question.length];
        System.arraycopy(header, 0, dnsRequest, 0, header.length);
        System.arraycopy(question, 0, dnsRequest, header.length, question.length);

        return dnsRequest;
    }
}
```

Output :



```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab>javac SimpleDNSClient.java

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab>java SimpleDNSClient
Raw DNS response:
AA AA 81 80 00 01 00 02 00 00 00 00 03 77 77 77 0A 43 6C 6F 75 64 66 6C 61 72 65 03 63 6F 6D 00 00 01 00 01 C0 0C 00 01
00 01 00 00 00 29 00 04 68 10 7B 60 C0 0C 00 01 00 01 00 00 00 29 00 04 68 10 7C 60

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab>
```

Result : Thus to write codes to study the basic simulation of DNS using UDP sockets has been initiated successfully.

EX NO: 5	WRITE CODES TO STIMULATE ARP/RARP PROTOCOLS.
DATE:	

Aim: To write codes to stimulate ARP/RARP protocols.

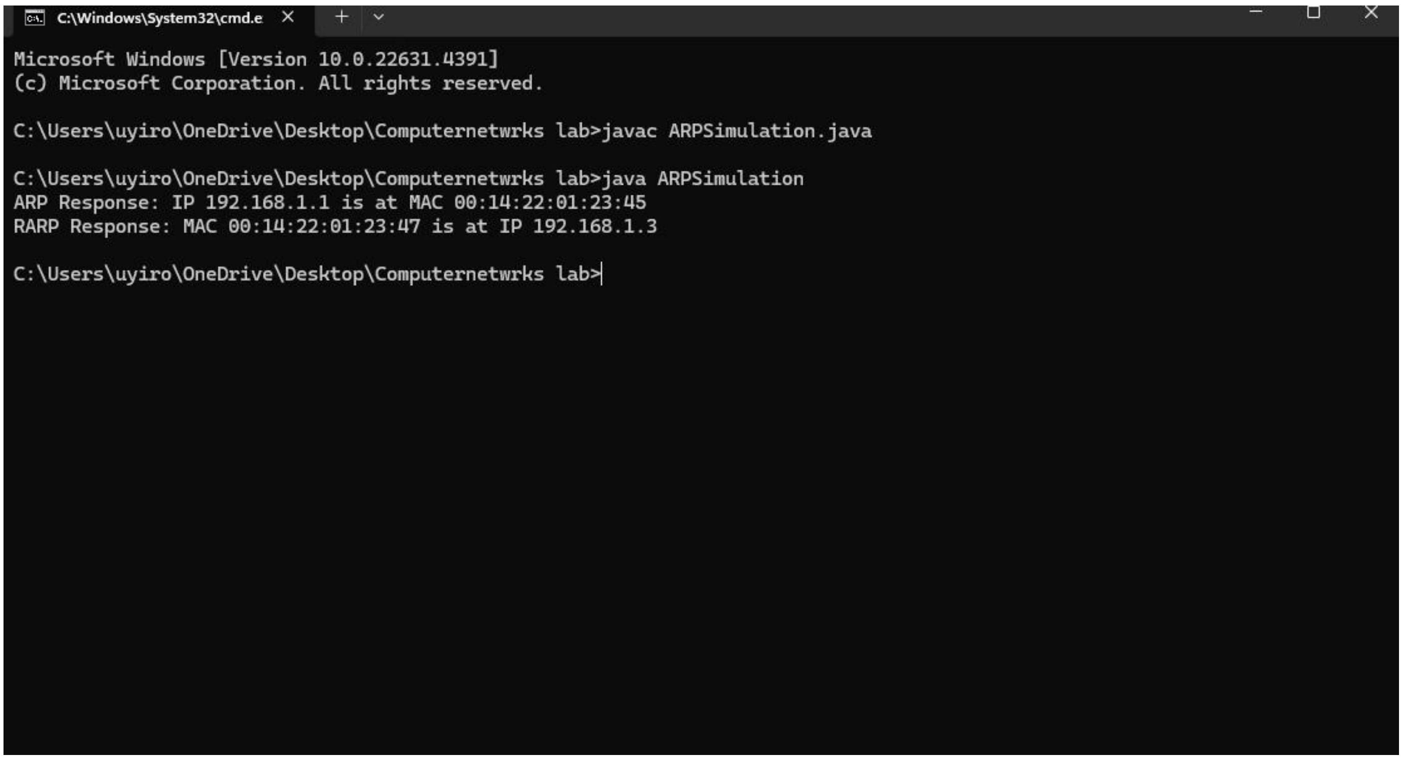
Algorithm:

1. Import the necessary java packages.
2. Recognize the mapping co-ordinates.
3. Record the request and response.
4. Reamplify the protocol addresses.
5. Observe the output and cross examine the codes.

Program:

```
import java.util.HashMap;
import java.util.Map;

public class ARPSimulation {

    private static Map<String, String> arpTable = new HashMap<>();

    static {

        arpTable.put("192.168.1.1", "00:14:22:01:23:45");
        arpTable.put("192.168.1.2", "00:14:22:01:23:46");
        arpTable.put("192.168.1.3", "00:14:22:01:23:47");
    }

    public static void main(String[] args) {

        String ipToResolve = "192.168.1.1";
        String macToResolve = "00:14:22:01:23:47";


        String macAddress = arpRequest(ipToResolve);
        if (macAddress != null) {
            System.out.println("ARP Response: IP " + ipToResolve + " is at MAC " +
macAddress);
        } else {
            System.out.println("ARP Response: IP " + ipToResolve + " is not found.");
        }
        String ipAddress = rarpRequest(macToResolve);
        if (ipAddress != null) {
            System.out.println("RARP Response: MAC " + macToResolve + " is at IP " +
ipAddress);
        } else {
            System.out.println("RARP Response: MAC " + macToResolve + " is not found.");
        }
    }


    private static String arpRequest(String ipAddress) {
        return arpTable.get(ipAddress);
    }


    private static String rarpRequest(String macAddress) {
        for (Map.Entry<String, String> entry : arpTable.entrySet()) {
            if (entry.getValue().equals(macAddress)) {
                return entry.getKey();
            }
        }
        return null;
    }
}
```

OUTPUT :



```
Microsoft Windows [Version 10.0.22631.4391]
(c) Microsoft Corporation. All rights reserved.

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab>javac ARPSimulation.java

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab>java ARPSimulation
ARP Response: IP 192.168.1.1 is at MAC 00:14:22:01:23:45
RARP Response: MAC 00:14:22:01:23:47 is at IP 192.168.1.3

C:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab>
```

Result : Thus to write codes to stimulate ARP/RARP protocols has been done successfully.

EX NO: 6	STUDY OF NETWORK SIMULATORS AND CONGESTION CONTROL ALGORITHM
DATE:	

Aim : To Study Of Network Simulators And Congestion Control Algorithm.

Algorithm:

1. Ns runs in both Windows, Linux .Highly preferred for easy control is linux(ubuntu).
2. Open the terminal on linux update the system , also install libraries , packages for firther execution.
3. Open up the text editor and start the coding with .tcl extension.
4. Compile and run the code which generates the trace file with .tr Extension.
5. Create a new graph file and feed your code , then save it with .gp extension.
6. Run the code the to view the graphical structure as your output.

Commands :

```
sudo apt update
sudo apt install ns2 nam gnuplot
gedit cs.tcl
ns cs.tcl
ls
gedit cs.tr
grep -e ‘^\+|^\-|^r’ cs.tr > filtered_tr.tr
ls
cat filtered_tr.tr | head -n 20
gedit ps.gp
gnuplot ps.gp
ls
eog pf.png
```

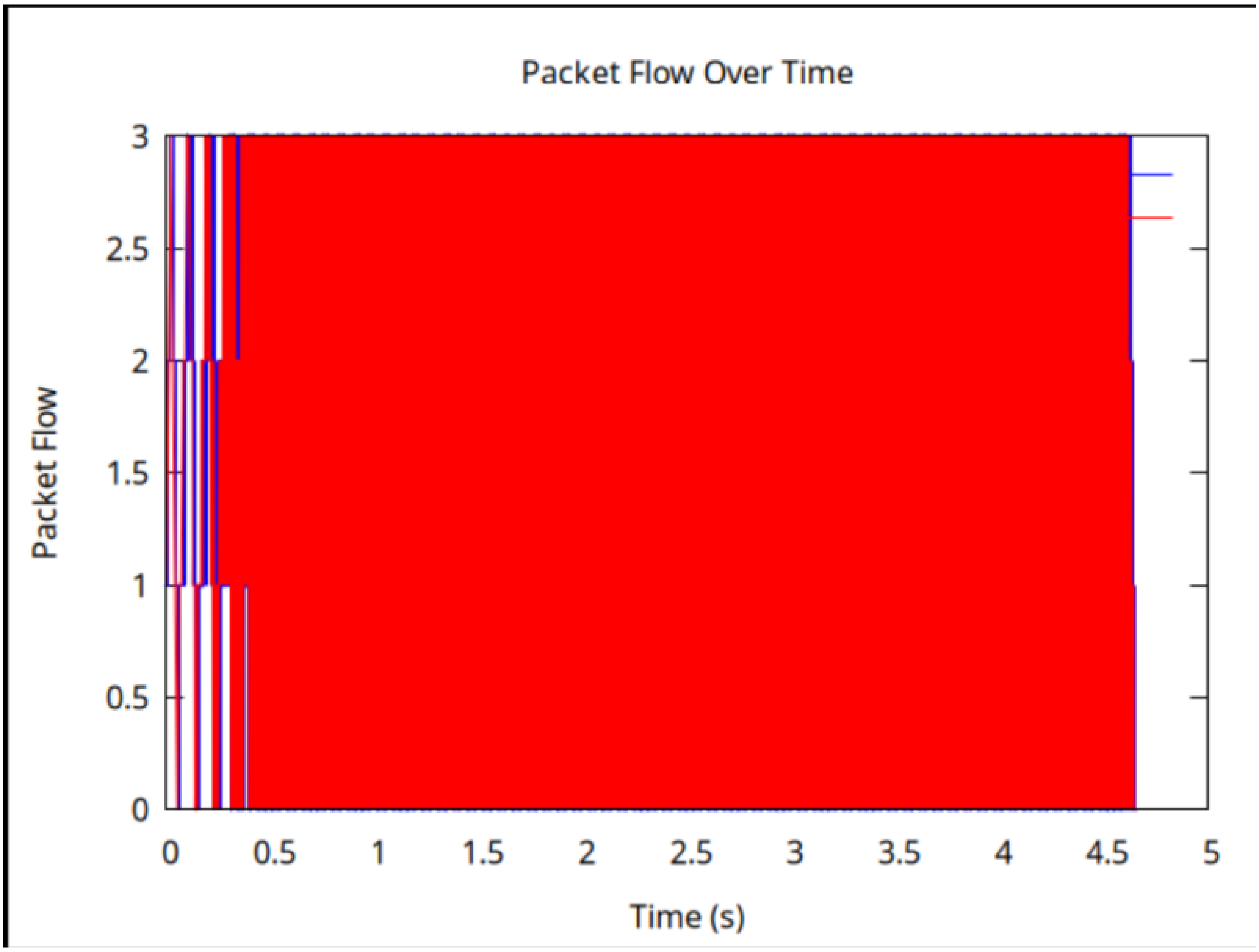
Program :
Extension : .tcl

```
set ns [new Simulator]

set tracefile [open CCS.tr w]
$ns trace-all $tracefile

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
$ns duplex-link $n2 $n3 1Mb 10ms DropTail

set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp

set sink [new Agent/TCPSink]
$ns attach-agent $n3 $sink

$ns connect $tcp $sink

set ftp [new Application/FTP]
$ftp attach-agent $tcp

$ns at 0.0 "$ftp start"
$ns at 4.5 "$ftp stop"
$ns at 5.0 "finish"

proc finish {} {
    global ns tracefile
    $ns flush-trace
    close $tracefile
    exit 0
}

$ns run
```

Extension : .gp

```
set terminal pngcairo
set output 'packet_flow.png'
set title 'Packet Flow Over Time'
set xlabel 'Time (s)'
set ylabel 'Packet Flow'
set xrange [0:*]
set yrange [0:*]
plot "valid_data.tr" using ($2):($4) with lines title 'Packet Flow (Sent)' linecolor rgb 'blue', \
     "valid_data.tr" using ($2):($4) with lines title 'Packet Flow (Received)' linecolor rgb 'red'
```

Output :



Result : Thus the study of network simulators and congestion control algorithm has been executed successfully.

EX NO: 7	STUDY OF TCP/UDP PERFORMANCE USING SIMULATION TOOL
DATE:	

Aim : To study the tcp/udp performance using simulation tools using ns2.

Algorithm:

1. Ns runs in both Windows, Linux .Highly preferred for easy control is linux(ubuntu).
2. Open the terminal on linux update the system , also install libraries , packages for firther execution.
3. Open up the text editor and start the coding with .tcl extension.
4. Compile and run the code which generates the trace file with .tr Extension &.awk extension.
5. Create a new graph file and feed your code , then save it with .gp extension.
6. Run the code the to view the graphical structure as your output.

Commands :

```
Gedit ex7tcp_udp.tcl
Ns ex7tcp_udp.tcl
Ls
Gedit trace.awk
Awk -f analyze_trace.awk congestion_simulation.tr > analyzed_trace.txt
Gedit trace.gp
Gnuplot trace.gp
Eog traffic_flow.png
```

Program :
Extension : .tcl

```
set ns [new Simulator]


set tracefile [open congestion_simulation.tr w]
$ns trace-all $tracefile

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 1Mb 10ms DropTail
```

```
set tcp [new Agent/TCP]
$ns attach-agent $n0 $tcp
set udp [new Agent/UDP]
$ns attach-agent $n0 $udp

set sink [new Agent/TCPSink]
$ns attach-agent $n2 $sink
set udpsink [new Agent/Null]
$ns attach-agent $n2 $udpsink

$ns connect $tcp $sink

$ns connect $udp $udpsink

set ftp [new Application/FTP]
$ftp attach-agent $tcp

$ns at 0.5 "$ftp start"
$ns at 4.5 "$ftp stop"

set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set packetSize_ 1000
$cbr set interval_ 0.1

$ns at 1.0 "$cbr start"
$ns at 4.0 "$cbr stop"


$ns at 10.0 "finish"

proc finish {} {
   global ns tracefile
   $ns flush-trace
   close $tracefile
   exit 0
}

$ns run
```

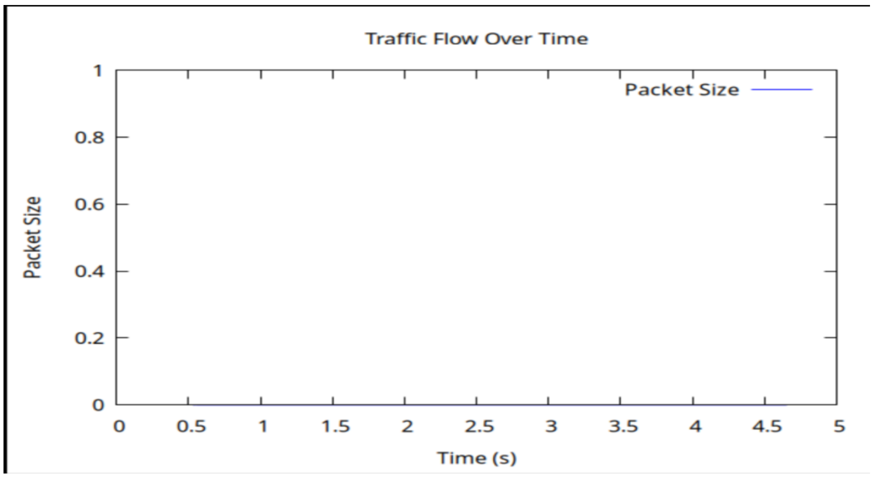
Extension : .awk

```
BEGIN {
   print "Time\tSource\tDest\tType\tSize"
}
{
   if ($1 == "+" || $1 == "-") {
        printf("%s\t%s\t%s\t%s\t%s\n", $2, $3, $4, $7, $8)
    }
}
```

Extension : .gp

```
set terminal pngcairo
set output 'traffic_flow.png'
set title 'Traffic Flow Over Time'
set xlabel 'Time (s)'
set ylabel 'Packet Size'
set xrange [0:*]
set yrange [0:*]
plot "analyzed_trace.txt" using 1:5 with lines title 'Packet Size' linecolor rgb 'blue'
```

Output :



Result : Thus to study the tcp/udp performance using simulation tools using ns2 has been done successfully

EX NO: 8a	SIMULATION OF LINK STATE ROUTING ALGORITHM
DATE:	

Aim : To write codes to simulate LSR.

Algorithm:

1. Ns runs in both Windows, Linux .Highly preferred for easy control is linux(ubuntu).
2. Open the terminal on linux update the system , also install libraries , packages for firther execution.
3. Open up the text editor and start the coding with .tcl extension.
4. Compile and run the code which generates the trace file with .tr Extension &.awk extension.
5. Create a new graph file and feed your code , then save it with .gp extension.
6. Run the code the to view the graphical structure as your output.

Commands :

```
gedit lsr.tcl
ns lsr.tcl
ls
gedit lsr.awk
awk -f lsr.awk dijkstra_trace.tr > lsr traced.txt
gedi# Create a simulator instance
 lsr.gp
gnuplot lsr.gp
```

Program :
Extension : .tcl

```
set ns [new Simulator]

set tracefile [open dijkstra_trace.tr w]
$ns trace-all $tracefile

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]

$ns duplex-link $n0 $n1 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 5ms DropTail
$ns duplex-link $n2 $n3 2Mb 5ms DropTail
```

```
$ns duplex-link $n3 $n4 2Mb 10ms DropTail
$ns duplex-link $n0 $n4 2Mb 20ms DropTail
$ns duplex-link $n2 $n4 2Mb 5ms DropTail

set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set null0 [new Agent/Null]
$ns attach-agent $n4 $null0

$ns connect $udp0 $null0

set cbr [new Application/Traffic/CBR]
$cbr set packetSize_ 512
$cbr set interval_ 0.2
$cbr attach-agent $udp0

$ns at 1.0 "$cbr start"
$ns at 4.0 "$cbr stop"

$ns at 5.0 "finish"

proc finish {} {
    global ns tracefile
    $ns flush-trace
    close $tracefile
    exit 0
}

$ns run
```

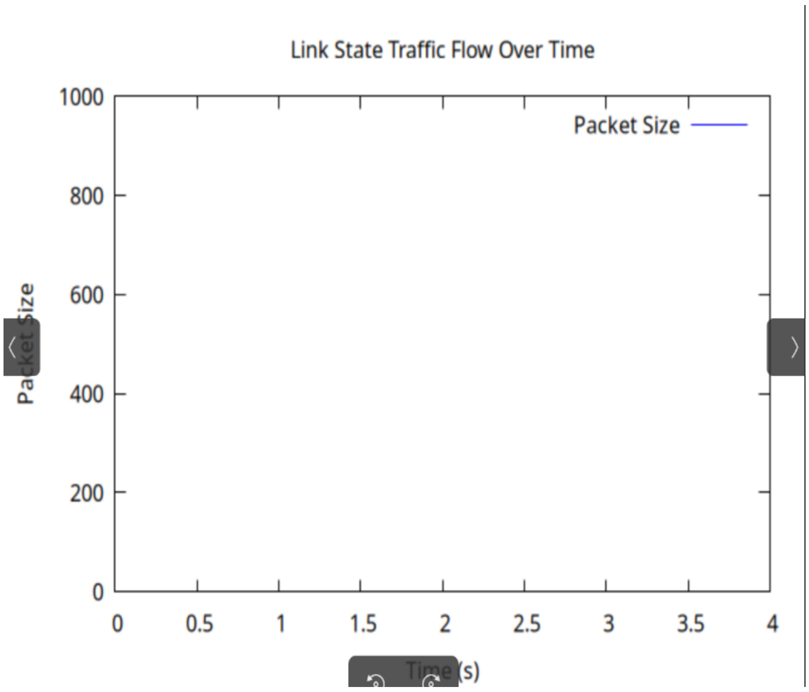
Extension : .awk

```
BEGIN {
    print "Time\tSource\tDest\tType\tSize"
}
{
    if ($1 == "+" || $1 == "-") {
        printf("%s\t%s\t%s\t%s\t%s\n", $2, $3, $4, $7, $8)
    }
}
```

Extension : .gp

```
set terminal pngcairo
set output 'LSR.png'
set title 'Link State Traffic Flow Over Time'
set xlabel 'Time (s)'
set ylabel 'Packet Size'
set xrange [0:*]
set yrange [0:1000]
plot "LSRtraced.txt" using 1:5 with lines title 'Packet Size' linecolor rgb 'blue'
```

Output :



Result : Thus to write codes to simulate LSR has been executed successfully.

EX NO: 8b	SIMULATION OF DISTANCE VECTOR ROUTING ALGORITHMS
DATE:	

Aim : To write codes to simulate DVR.

Algorithm:

1. Ns runs in both Windows, Linux .Highly preferred for easy control is linux(ubuntu).
2. Open the terminal on linux update the system , also install libraries , packages for firther execution.
3. Open up the text editor and start the coding with .tcl extension.
4. Compile and run the code which generates the trace file with .tr Extension &.awk extension.
5. Create a new graph file and feed your code , then save it with .gp extension.
6. Run the code the to view the graphical structure as your output.

Commands :

```
gedit dvr.tcl
ns dvr.tcl
ls
gedit 8btrace.awk
awk -f 8btrace.awk dvr_trace.tr >  8btraced.txt
gedit 8btrace.gp
gnuplot 8btrace.gp
eog dvr.png
```

Program :
Extension : .tcl

```
set ns [new Simulator]

set tracefile [open dvr_trace.tr w]
$ns trace-all $tracefile

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
```

```
$ns duplex-link $n0 $n1 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 2Mb 10ms DropTail
$ns duplex-link $n3 $n4 2Mb 10ms DropTail


set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set null0 [new Agent/Null]
$ns attach-agent $n4 $null0

$ns connect $udp0 $null0

set cbr [new Application/Traffic/CBR]
$cbr set packetSize_ 512
$cbr set interval_ 0.2
$cbr attach-agent $udp0

$ns at 1.0 "$cbr start"
$ns at 4.0 "$cbr stop"

$ns at 5.0 "finish"

proc finish {} {
    global ns tracefile
    $ns flush-trace
    close $tracefile
    exit 0
}

$ns run
[[
```

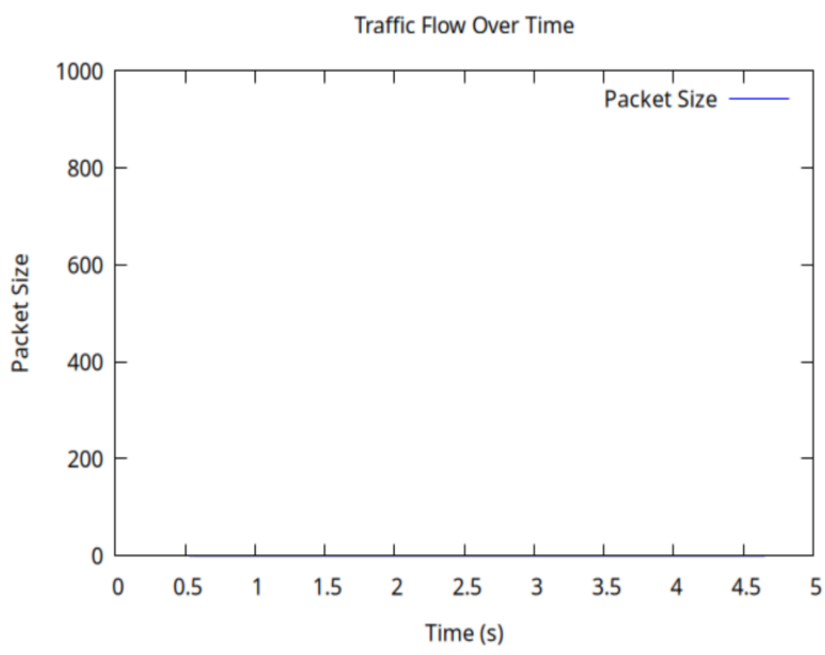
Extension : .awk

```
BEGIN {
    print "Time\tSource\tDest\tType\tSize"
}
{
    if ($1 == "+" || $1 == "-") {
        printf("%s\t%s\t%s\t%s\t%s\n", $2, $3, $4, $6, $8)
    }
}
```

Extension : .gp

```
set terminal pngcairo
set output 'DVR.png'
set title 'Traffic Flow Over Time'
set xlabel 'Time (s)'
set ylabel 'Packet Size'
set xrange [0:*]
set yrange [0:1000]
plot "analyzed_trace.txt" using 1:5 with lines title 'Packet Size' linecolor rgb 'blue'
```

Output :



Result : Thus to write codes to simulate DVR has been executed successfully.

EX NO: 9	UNICAST ROUTING PROTOCOL
DATE:	

Aim : To write codes to simulate unicast routing protocol.

Algorithm:

1. Ns runs in both Windows, Linux .Highly preferred for easy control is linux(ubuntu).
2. Open the terminal on linux update the system , also install libraries , packages for firther execution.
3. Open up the text editor and start the coding with .tcl extension.
4. Compile and run the code which generates the trace file with .tr Extension &.awk extension.
5. Create a new graph file and feed your code , then save it with .gp extension.
6. Run the code the to view the graphical structure as your output.

Commands :

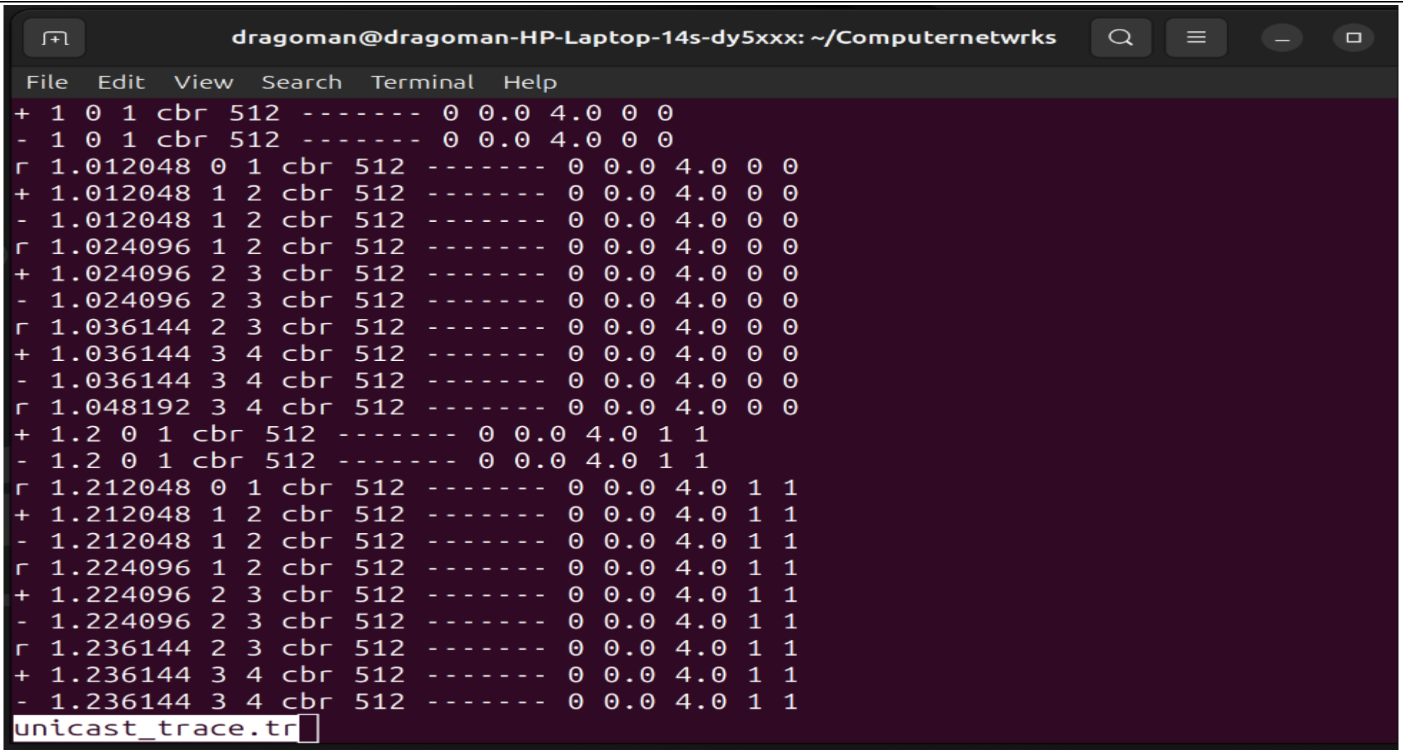
```
gedit ex9.tcl
ns ex9.tcl
Less unicast_trace.tr
q
```

Program :

```
set ns [new Simulator]


set tracefile [open unicast_trace.tr w]
$ns trace-all $tracefile


set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
```

```
$ns duplex-link $n0 $n1 2Mb 10ms DropTail
$ns duplex-link $n1 $n2 2Mb 10ms DropTail
$ns duplex-link $n2 $n3 2Mb 10ms DropTail
$ns duplex-link $n3 $n4 2Mb 10ms DropTail


set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set null0 [new Agent/Null]
$ns attach-agent $n4 $null0


$ns connect $udp0 $null0


set cbr [new Application/Traffic/CBR]
$cbr set packetSize_ 512
$cbr set interval_ 0.2
$cbr attach-agent $udp0



$ns at 1.0 "$cbr start"
$ns at 4.0 "$cbr stop"


$ns at 5.0 "finish"


proc finish {} {
   global ns tracefile
   $ns flush-trace
   close $tracefile
   exit 0
}


$ns run
```

Output :


dragoman@dragoman-HP-Laptop-14s-dy5xxx: ~/Computernetwrks
File Edit View Search Terminal Help

```
+ 1 0 1 cbr 512 ------- 0 0.0 4.0 0 0
- 1 0 1 cbr 512 ------- 0 0.0 4.0 0 0
r 1.012048 0 1 cbr 512 ------- 0 0.0 4.0 0 0
+ 1.012048 1 2 cbr 512 ------- 0 0.0 4.0 0 0
- 1.012048 1 2 cbr 512 ------- 0 0.0 4.0 0 0
r 1.024096 1 2 cbr 512 ------- 0 0.0 4.0 0 0
+ 1.024096 2 3 cbr 512 ------- 0 0.0 4.0 0 0
- 1.024096 2 3 cbr 512 ------- 0 0.0 4.0 0 0
r 1.036144 2 3 cbr 512 ------- 0 0.0 4.0 0 0
+ 1.036144 3 4 cbr 512 ------- 0 0.0 4.0 0 0
- 1.036144 3 4 cbr 512 ------- 0 0.0 4.0 0 0
r 1.048192 3 4 cbr 512 ------- 0 0.0 4.0 0 0
+ 1.2 0 1 cbr 512 ------- 0 0.0 4.0 1 1
- 1.2 0 1 cbr 512 ------- 0 0.0 4.0 1 1
r 1.212048 0 1 cbr 512 ------- 0 0.0 4.0 1 1
+ 1.212048 1 2 cbr 512 ------- 0 0.0 4.0 1 1
- 1.212048 1 2 cbr 512 ------- 0 0.0 4.0 1 1
r 1.224096 1 2 cbr 512 ------- 0 0.0 4.0 1 1
+ 1.224096 2 3 cbr 512 ------- 0 0.0 4.0 1 1
- 1.224096 2 3 cbr 512 ------- 0 0.0 4.0 1 1
r 1.236144 2 3 cbr 512 ------- 0 0.0 4.0 1 1
+ 1.236144 3 4 cbr 512 ------- 0 0.0 4.0 1 1
- 1.236144 3 4 cbr 512 ------- 0 0.0 4.0 1 1
unicast_trace.tr
```

Result : Thus to write codes to simulate UCR has been executed successfully.

EX NO: 10	SIMULATION OF ERROR CORRECTION CODE (CRC)
DATE:	

Aim : To write codes to simulate error correction codes like CRC.

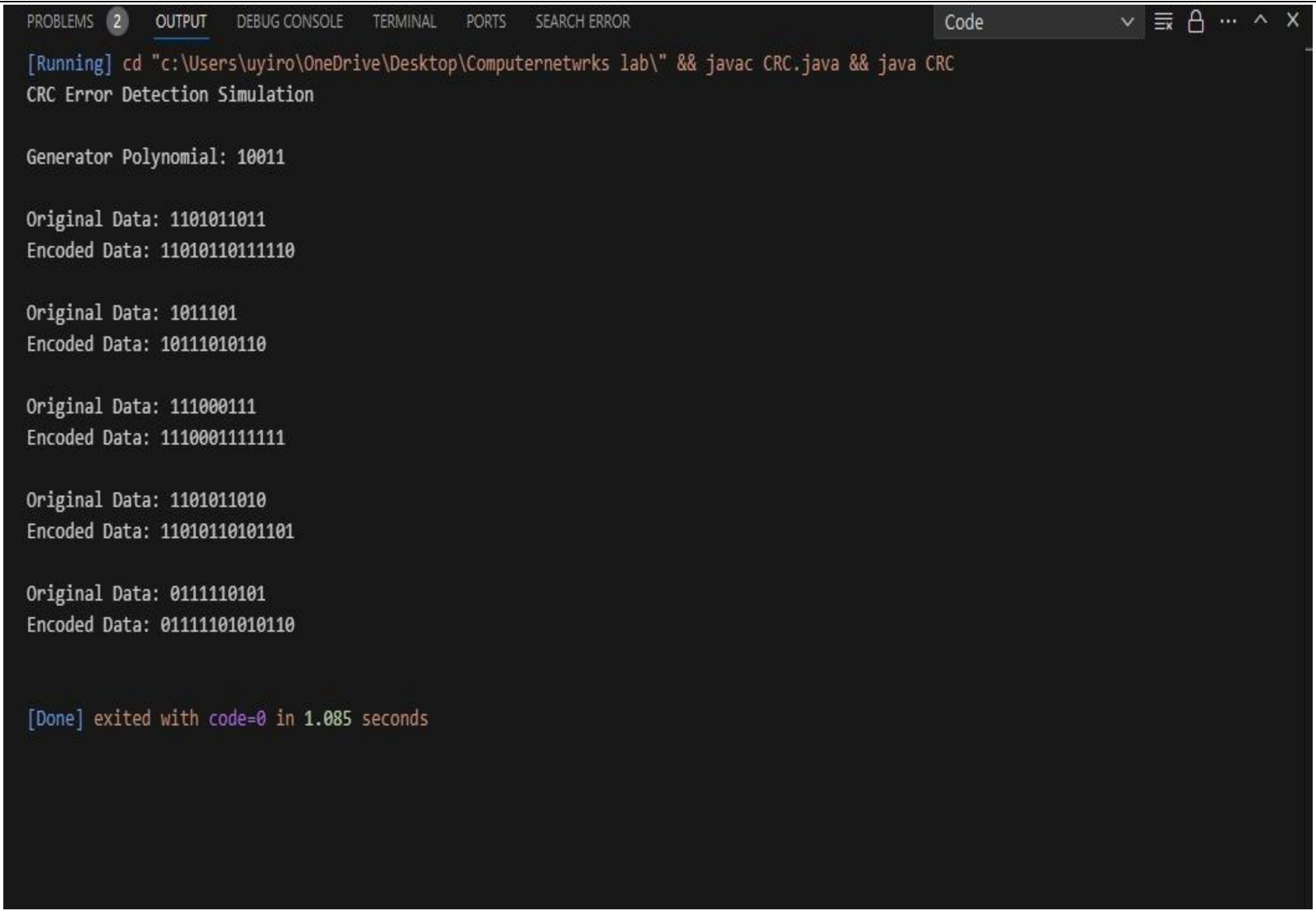
Algorithm:

1. Import necessary java utils.
2. By using notepad, vsc you can structure up the codes .
3. Feed the datasets to codes.
4. Execute it using java commands.
5. Further data updates lead to the great study of ECC.
6. Compile and run the code to observe the outputs.

Program :

```
import java.util.Arrays;

public class CRC {
    public static void main(String[] args) {
        String[] dataSet = {"1101011011", "1011101", "111000111", "1101011010",
"0111110101"};
        String generator = "10011";

        System.out.println("CRC Error Detection Simulation\n");
        System.out.println("Generator Polynomial: " + generator + "\n");

        for (String data : dataSet) {
            String encodedData = encodeData(data, generator);
            System.out.println("Original Data: " + data);
            System.out.println("Encoded Data: " + encodedData + "\n");
        }
    }

    // Function to perform XOR operation
    static String xor(String a, String b) {
        StringBuilder result = new StringBuilder();
        for (int i = 1; i < b.length(); i++) {
            if (a.charAt(i) == b.charAt(i))
                result.append("0");
            else
                result.append("1");
        }
        return result.toString();
    }

    // Function to perform Modulo-2 division
    static String mod2div(String dividend, String divisor) {
        int pick = divisor.length();
        String tmp = dividend.substring(0, pick);
        int n = dividend.length();
        while (pick < n) {
            if (tmp.charAt(0) == '1')
                tmp = xor(divisor, tmp) + dividend.charAt(pick);
            else
                tmp = xor("0".repeat(pick), tmp) + dividend.charAt(pick);
            pick += 1;
        }
        if (tmp.charAt(0) == '1')
            tmp = xor(divisor, tmp);
        else
            tmp = xor("0".repeat(pick), tmp);
        return tmp;
    }

    // Function to encode data using CRC
    static String encodeData(String data, String generator) {
        int dataLen = data.length();
        int generatorLen = generator.length();
        String appendedData = data + "0".repeat(generatorLen - 1);
        String remainder = mod2div(appendedData, generator);
        return data + remainder;
    }
}
```

Output :



```
[Running] cd "c:\Users\uyiro\OneDrive\Desktop\Computernetwrks lab\" && javac CRC.java && java CRC
CRC Error Detection Simulation

Generator Polynomial: 10011

Original Data: 1101011011
Encoded Data: 11010110111110

Original Data: 1011101
Encoded Data: 10111010110

Original Data: 111000111
Encoded Data: 1110001111111

Original Data: 1101011010
Encoded Data: 11010110101101

Original Data: 0111110101
Encoded Data: 01111101010110


[Done] exited with code=0 in 1.085 seconds
```

Result : Thus to write codes to simulate ECC has been executed successfully.