1. Implement three nodes point –to-point network with duplex links between them. Set the queue size, vary the bandwidth, and find the number of packets dropped.

Commands

gedit lab1.tcl

```
set ns [new Simulator]
set nf [open lab1.nam w]
$ns namtrace-all $nf
set nd [open lab1.tr w]
$ns trace-all $nd
proc finish {} {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam lab1.nam &
exit 0
}
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
$ns duplex-link $n0 $n1 1Mb 10ms DropTail
$ns duplex-link $n1 $n2 512kb 10ms DropTail
$ns queue-limit $n1 $n2 10
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
```

$cbr0 set interval_ 0.005

$cbr0 attach-agent $udp0

set sink [new Agent/Null]

$ns attach-agent $n2 $sink

$ns connect $udp0 $sink

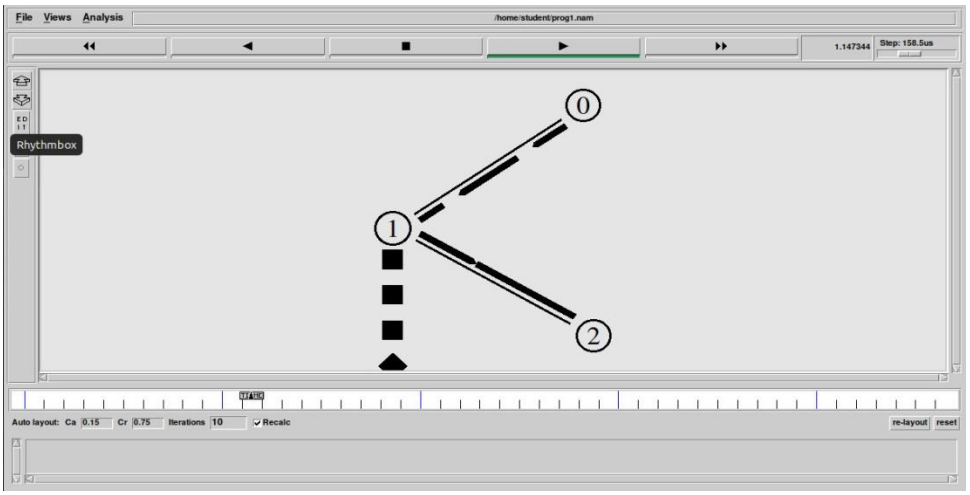
$ns at 0.2 "$cbr0 start"

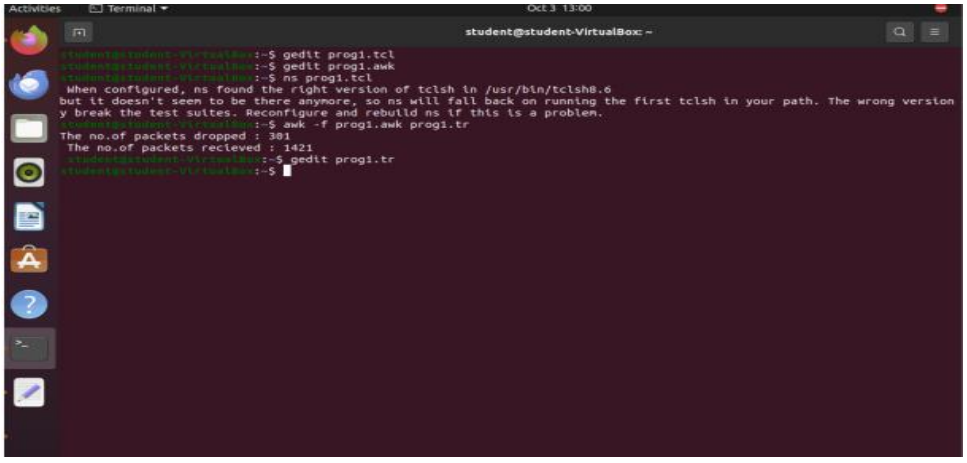
$ns at 4.5 "$cbr0 stop"

$ns at 5.0 "finish"

$ns run

ns lab1.tcl





gedit program1.awk

```
BEGIN{
dcount=0;
rcount=0;
}{
event=$1;
if(event=="d")
{
dcount++;
}
if(event=="r")
{
rcount++;
}
}
END{
printf("The no. of packets dropped: %d\n",dcount);
printf("The no. of packets reveived: %d\n",rcount);
}
```

awk -f program1.awk lab1.tr

OUTPUT:

The no. of packets dropped: 10

The no. of packets received: 200

2. Implement transmission of ping messages/trace route over a network topology consisting of 6 nodes and find the number of packets dropped due to congestion.

gedit prg2.tcl

```
#Create Simulator
set ns [new Simulator]

#Use colors to differentiate the traffic
$ns color 1 Blue
$ns color 2 Red

#Open trace and NAM trace file
set ntrace [open prg2.tr w]
$ns trace-all $ntrace
set namfile [open prg2.nam w]
$ns namtrace-all $namfile

#Finish Procedure
proc Finish {} {
global ns ntrace namfile
#Dump all trace data and close the file
$ns flush-trace
close $ntrace
close $namfile

#Execute the nam animation file
exec nam prg2.nam &
```

```
#Find the number of ping packets dropped
puts "The number of ping packets dropped are "
exec grep "^d" prg2.tr | cut .d " " .f s | grep .c "ping" &
exit 0
}

#Create six nodes
for {set i 0} {$i < 6} {incr i} {
set n($i) [$ns node]
}

#Connect the nodes
for {set j 0} {$j < 5} {incr j} {
$ns duplex-link $n($j) $n([expr ($j+1)]) 0.1Mb 10ms DropTail
}

#Define the recv function for the class 'Agent/Ping'
Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts "node [$node_ id] received ping answer from $from with round trip time $rtt
ns"
}

#Create two ping agents and attach them to n(0) and n(5)
set p0 [new Agent/Ping]
$p0 set class_ 1
$ns attach-agent $n(0) $p0
```

```
set p1 [new Agent/Ping]
set class_ 1
$ns attach-agent $n(5) $p1
$ns connect $p0 $p1

#Set queue size and monitor the queue
#Queue size is set to 2 to observe the drop in ping packets
$ns queue-limit $n(2) $n(3) 2
$ns duplex-link-op $n(2) $n(3) queuePos 0.5

#Create Congestion
#Generate a Huge CBR traffic between n(2) and n(4)
set tcp0 [new Agent/TCP]
$tcp0 set class_ 2
$ns attach-agent $n(2) $tcp0
set sink0 [new Agent/TCPSink]
$ns attach-agent $n(4) $sink0
$ns connect $tcp0 $sink0

#Apply CBR traffic over TCP
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set rate_ 1Mb
$cbr0 attach-agent $tcp0

#Schedule events
$ns at 0.2 "$p0 send"
```

$ns at 0.4 "$p1 send"

$ns at 0.4 "$cbr0 start"

$ns at 0.8 "$p0 send"

$ns at 1.0 "$p1 send"

$ns at 1.2 "$cbr0 stop"

$ns at 1.4 "$p0 send"

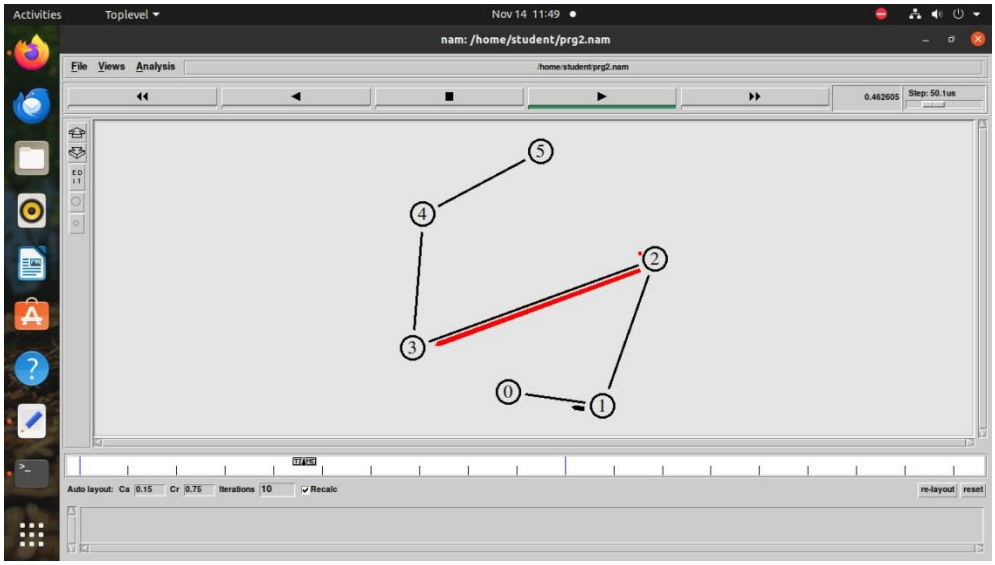
$ns at 1.6 "$p1 send"

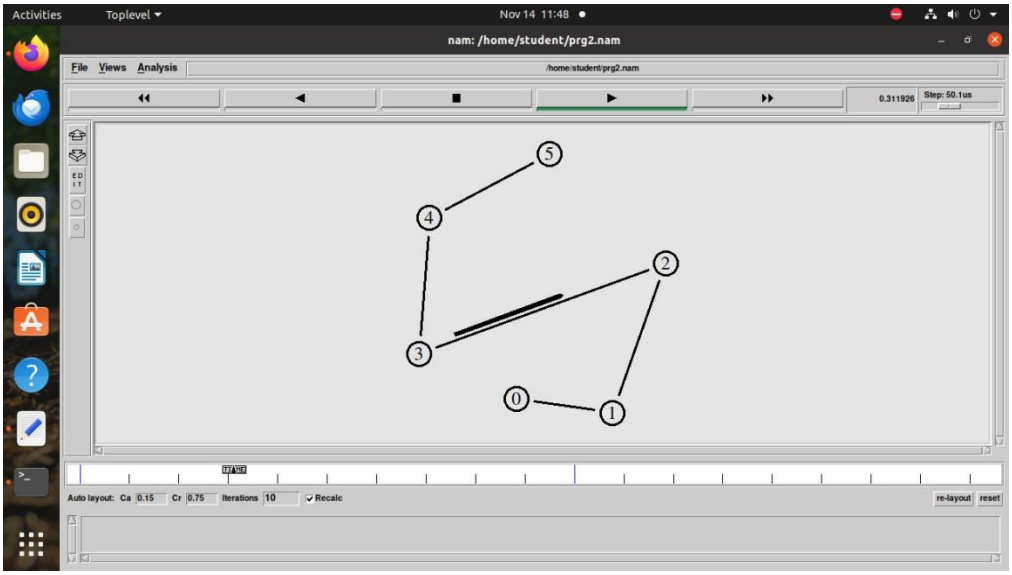
$ns at 1.8 "Finish"

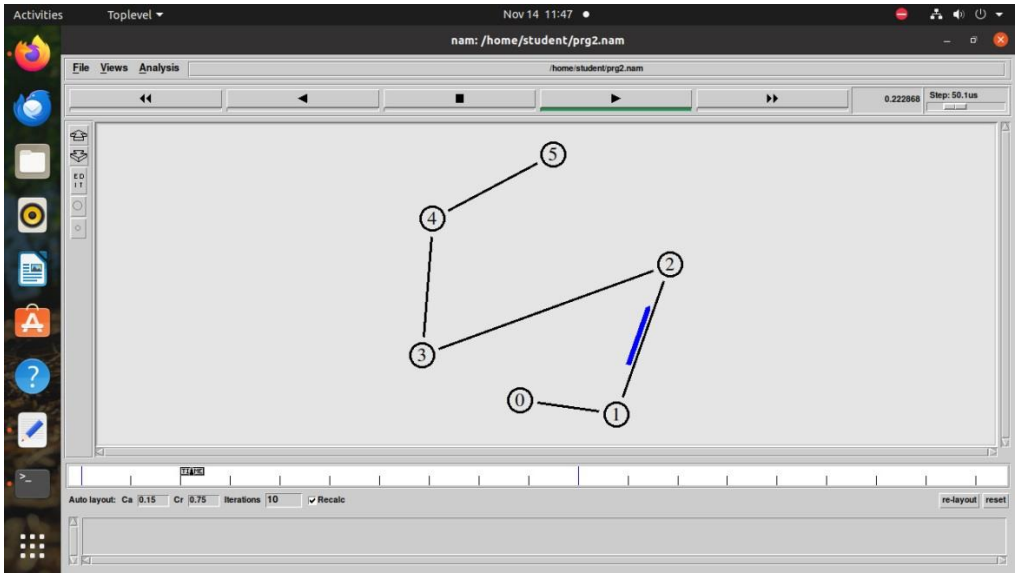
$ns run

ns prg2.tcl

OUTPUT




Activities
Toplevel
Nov 14 11:48
nam: /home/student/prg2.nam
File Views Analysis
/home/student/prg2.nam
0.311926
Step: 50.1us
5
4
2
3
0
1
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout reset


Activities
Toplevel
Nov 14 11:47
nam: /home/student/prg2.nam
File Views Analysis
/home/student/prg2.nam
0.222868
Step: 50.1us
5
4
2
3
0
1
Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc
re-layout reset

3. Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.

gedit prog3.tcl

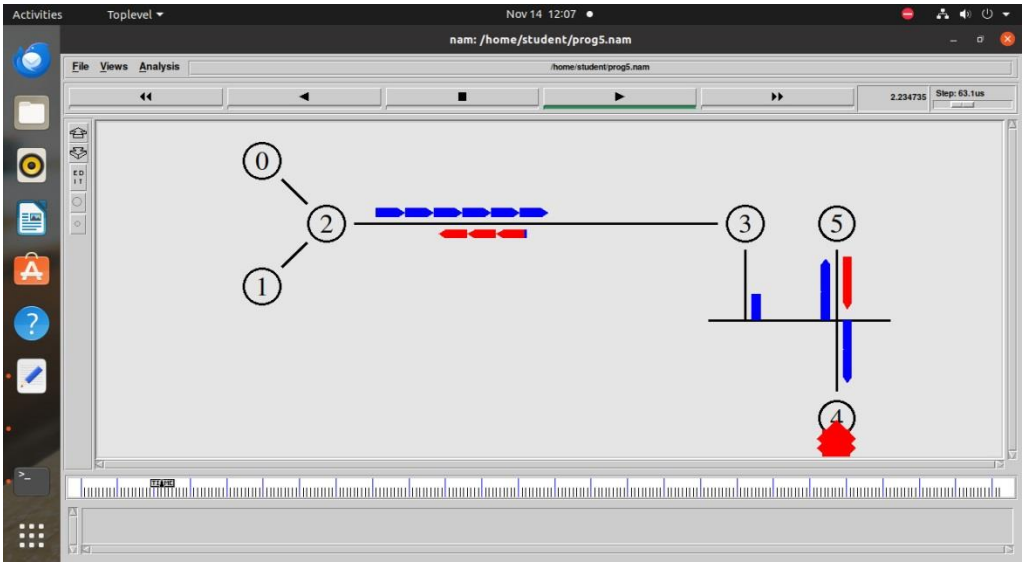
```
# Create Simulator
set ns [new Simulator]

# Use colors to differentiate the traffic
$ns color 1 Blue
$ns color 2 Red

# Open trace and NAM trace file
set ntrace [open prog5.tr w]
$ns trace-all $ntrace
set namfile [open prog5.nam w]
$ns namtrace-all $namfile

# Create flat files to create congestion graph windows
set winFile0 [open WinFile0 w]
set winFile1 [open WinFile1 w]

# Finish Procedure
proc Finish {} {
   # Dump all trace data and close the files
   global ns ntrace namfile
   $ns flush-trace
   close $ntrace
   close $namfile

   # Execute the NAM animation file
   exec nam prog5.nam &

   # Plot the Congestion Window graph using xgraph
   exec xgraph WinFile0 WinFile1 &
   exit 0
}

# Plot Window Procedure
proc PlotWindow {tcpSource file} {
   global ns
```

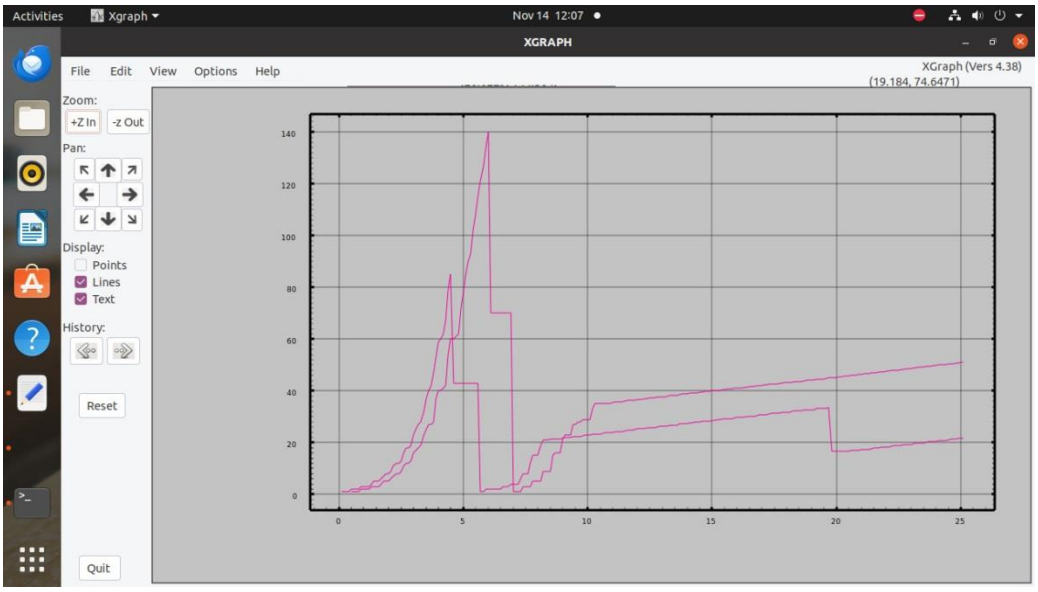
```
    set time 0.1
    set now [$ns now]

    set cwnd [$tcpSource set cwnd_]
    puts $file "$now $cwnd"

    $ns at [expr $now+$time] "PlotWindow $tcpSource $file"
}

# Create 6 nodes
for {set i 0} {$i < 6} {incr i} {
    set n($i) [$ns node]
}

# Create duplex links between the nodes
$ns duplex-link $n(0) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(1) $n(2) 2Mb 10ms DropTail
$ns duplex-link $n(2) $n(3) 0.6Mb 100ms DropTail

# Nodes n(3), n(4), and n(5) are considered in a LAN
set lan [$ns newLan "$n(3) $n(4) $n(5)" 0.5Mb 40ms LL Queue/DropTail MAC/802_3 Channel]

# Orientation to the nodes
$ns duplex-link-op $n(0) $n(2) orient right-down
$ns duplex-link-op $n(1) $n(2) orient right-up
$ns duplex-link-op $n(2) $n(3) orient right

# Setup queue between n(2) and n(3) and monitor the queue
$ns queue-limit $n(2) $n(3) 20
$ns duplex-link-op $n(2) $n(3) queuePos 0.5

# Set error model on link n(2) to n(3)
set loss_module [new ErrorModel]
$loss_module ranvar [new RandomVariable/Uniform]
$loss_module drop-target [new Agent/Null]
$ns lossmodel $loss_module $n(2) $n(3)

# Set up the TCP connection between n(0) and n(4)
set tcp0 [new Agent/TCP/Newreno]
$tcp0 set fid_ 1
```

```
$tcp0 set window_ 8000
$tcp0 set packetSize_ 552
$ns attach-agent $n(0) $tcp0
set sink0 [new Agent/TCPSink/DelAck]
$ns attach-agent $n(4) $sink0
$ns connect $tcp0 $sink0

# Apply FTP Application over TCP
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ftp0 set type_ FTP

# Set up another TCP connection between n(5) and n(1)
set tcp1 [new Agent/TCP/Newreno]
$tcp1 set fid_ 2
$tcp1 set window_ 8000
$tcp1 set packetSize_ 552
$ns attach-agent $n(5) $tcp1
set sink1 [new Agent/TCPSink/DelAck]
$ns attach-agent $n(1) $sink1
$ns connect $tcp1 $sink1

# Apply FTP application over TCP
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ftp1 set type_ FTP

# Schedule Events
$ns at 0.1 "$ftp0 start"
$ns at 0.1 "PlotWindow $tcp0 $winFile0"
$ns at 0.5 "$ftp1 start"
$ns at 0.5 "PlotWindow $tcp1 $winFile1"
$ns at 25.0 "$ftp0 stop"
$ns at 25.1 "$ftp1 stop"
$ns at 25.2 "Finish"

# Run the simulation
$ns run
```

ns prog3.tcl

OUTPUT





4. Develop a program for error detecting code using CRC-CCITT (16- bits).

nano CRC4.java

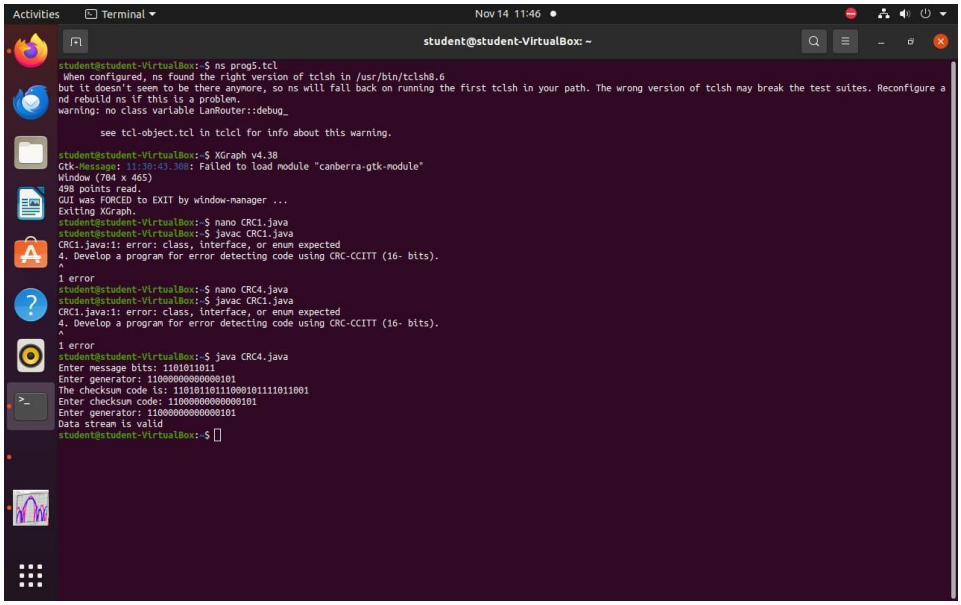
```
import java.util.Scanner;
import java.io.*;
public class CRC4 {

   public static void main(String args[]) {

   Scanner sc = new Scanner(System.in);

   //Input Data Stream
   System.out.print("Enter message bits: ");
   String message = sc.nextLine();
   System.out.print("Enter generator: ");
   String generator = sc.nextLine();
int data[] = new int[message.length() + generator.length() - 1];
int divisor[] = new int[generator.length()];
for(int i=0;i<message.length();i++)
        data[i] = Integer.parseInt(message.charAt(i)+"");
for(int i=0;i<generator.length();i++)
        divisor[i] = Integer.parseInt(generator.charAt(i)+"");

//Calculation of CRC
for(int i=0;i<message.length();i++)
{
        if(data[i]==1)
                for(int j=0;j<divisor.length;j++)
                        data[i+j] ^= divisor[j];
}

//Display CRC
System.out.print("The checksum code is: ");
for(int i=0;i<message.length();i++)
        data[i] = Integer.parseInt(message.charAt(i)+"");
for(int i=0;i<data.length;i++)
   System.out.print(data[i]);
System.out.println();

//Check for input CRC code
```

```
System.out.print("Enter checksum code: ");
        message = sc.nextLine();
System.out.print("Enter generator: ");
        generator = sc.nextLine();
data = new int[message.length() + generator.length() - 1];
divisor = new int[generator.length()];
for(int i=0;i<message.length();i++)
        data[i] = Integer.parseInt(message.charAt(i)+"");
for(int i=0;i<generator.length();i++)
        divisor[i] = Integer.parseInt(generator.charAt(i)+"");

//Calculation of remainder
for(int i=0;i<message.length();i++) {
        if(data[i]==1)
                for(int j=0;j<divisor.length;j++)
                        data[i+j] ^= divisor[j];
}

//Display validity of data
boolean valid = true;
for(int i=0;i<data.length;i++)
        if(data[i]==1){
                valid = false;
                break;
}

if(valid==true)
        System.out.println("Data stream is valid");
else
        System.out.println("Data stream is invalid. CRC error occurred.");
}

}
```

Press Ctrl + O (to write out).
Press Enter to confirm the file name.
Press Ctrl + X to exit nano.

```
javac CRC4.java
java CRC4
```

OUTPUT


Activities
Terminal
Nov 14 11:46
student@student-VirtualBox: ~
student@student-VirtualBox:~$ ns prog5.tcl
When configured, ns found the right version of tclsh in /usr/bin/tclsh8.6
but it doesn't seem to be there anymore, so ns will fall back on running the first tclsh in your path. The wrong version of tclsh may break the test suites. Reconfigure a
nd rebuild ns if this is a problem.
warning: no class variable LanRouter::debug_
see tcl-object.tcl in tclcl for info about this warning.
student@student-VirtualBox:~$ XGraph v4.38
Gtk-Message: 11:50:43.308: Failed to load module "canberra-gtk-module"
Window (704 x 465)
498 points read.
GUI was FORCED to EXIT by window-manager ...
Exiting XGraph.
student@student-VirtualBox:~$ nano CRC1.java
student@student-VirtualBox:~$ javac CRC1.java
CRC1.java:1: error: class, interface, or enum expected
4. Develop a program for error detecting code using CRC-CCITT (16- bits).
^
1 error
student@student-VirtualBox:~$ nano CRC4.java
student@student-VirtualBox:~$ javac CRC1.java
CRC1.java:1: error: class, interface, or enum expected
4. Develop a program for error detecting code using CRC-CCITT (16- bits).
^
1 error
student@student-VirtualBox:~$ java CRC4.java
Enter message bits: 1101011011
Enter generator: 1100000000000101
The checksum code is: 11010110111000101111011001
Enter checksum code: 11000000000000101
Enter generator: 1100000000000101
Data stream is valid
student@student-VirtualBox:~$

5. Develop a program to find the shortest path between vertices using the Bellman-Ford and path vector routing algorithm.

nano ford.java

```
import java.util.Scanner;
public class ford
{
 private int D[];
 private int num_ver;
 public static final int MAX_VALUE = 999;
  public ford(int num_ver)
  {
    this.num_ver = num_ver;
    D = new int[num_ver + 1];
  }

 public void BellmanFordEvaluation(int source, int A[][])
 {
  for (int node = 1; node <= num_ver; node++)
   {
       D[node] = MAX_VALUE;
   }

 D[source] = 0;

 for (int node = 1; node <= num_ver - 1; node++)
    {
      for (int sn = 1; sn <= num_ver; sn++)
      {
        for (int dn = 1; dn <= num_ver; dn++)
          {
            if (A[sn][dn] != MAX_VALUE)
            {
                    if (D[dn] > D[sn]+ A[sn][dn])
                            D[dn] = D[sn] + A[sn][dn];
            }
          }
        }
    }
```

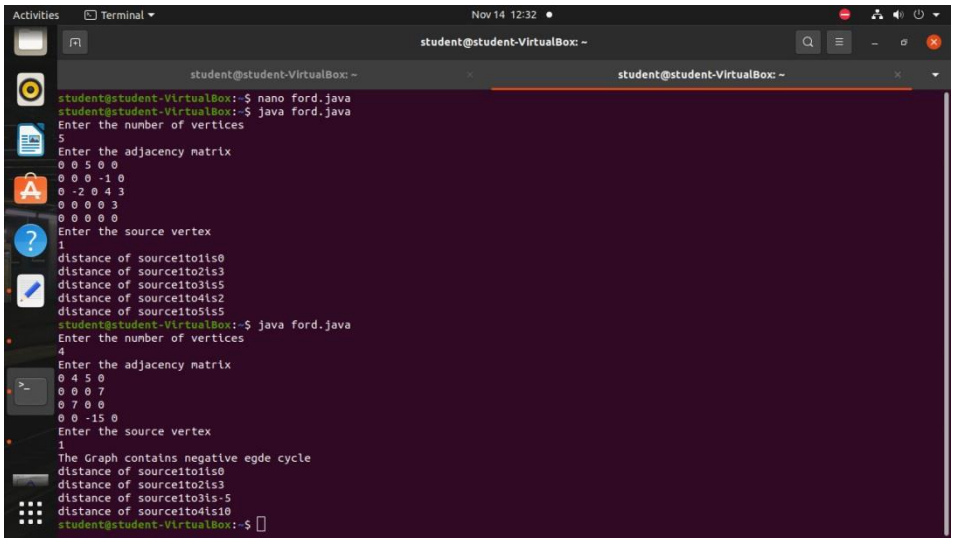
```
  for (int sn = 1; sn <= num_ver; sn++)
  {
   for (int dn = 1; dn <= num_ver; dn++)
    {
    if (A[sn][dn] != MAX_VALUE)
    {
          if (D[dn] > D[sn]+ A[sn][dn])
System.out.println("The Graph contains negative egde cycle");                    }
 }
}

  for (int vertex = 1; vertex <= num_ver; vertex++)
  {
System.out.println("distance of source"+source+"to"+vertex+"is" + D[vertex]);
   }
}

 public static void main(String[ ] args)
 {
   int num_ver = 0;
   int source;
   Scanner scanner = new Scanner(System.in);
   System.out.println("Enter the number of vertices");
    num_ver = scanner.nextInt();

    int A[][] = new int[num_ver + 1][num_ver + 1];
   System.out.println("Enter the adjacency matrix");
  for (int sn = 1; sn <= num_ver; sn++)
  {
   for (int dn = 1; dn <= num_ver; dn++)
   {
         A[sn][dn] = scanner.nextInt();
         if (sn == dn)
         {
         A[sn][dn] = 0;
         continue;
     }
         if (A[sn][dn] == 0)
         {
         A[sn][dn] = MAX_VALUE;
```

```
            }
            }
        }


            System.out.println("Enter the source vertex");
         source = scanner.nextInt();
        ford b = new ford (num_ver);
        b.BellmanFordEvaluation(source, A);
        scanner.close();
        }
}
```

Press Ctrl + O (to write out).
Press Enter to confirm the file name.
Press Ctrl + X to exit nano.

```
javac ford.java
java ford
```

OUTPUT



6. Using TCP/IP sockets, write a client – server program to make the client send the file name and to make the server send back the contents of the requested file if present.

nano TCPS.java

```
import java.net.*;
import java.io.*;
public class TCPS
{
public static void main(String[] args) throws Exception
{
ServerSocket sersock=new ServerSocket(4000);
System.out.println("Server ready for connection");

Socket sock=sersock.accept();

System.out.println("Connection Is successful and waiting for chatting");

InputStream istream=sock.getInputStream();

BufferedReader fileRead=new BufferedReader(new InputStreamReader(istream));

String fname=fileRead.readLine();

BufferedReader ContentRead=new BufferedReader(new FileReader(fname));

OutputStream ostream=sock.getOutputStream();

PrintWriter pwrite=new PrintWriter(ostream,true);

String str;

while((str=ContentRead.readLine())!=null){

pwrite.println(str);

}
sock.close();
sersock.close();
```

```
pwrite.close();
fileRead.close();
ContentRead.close();
}
}
```

Press Ctrl + O (to write out).
Press Enter to confirm the file name.
Press Ctrl + X to exit nano.
nano TCPC.java

```
import java.net.*;
import java.io.*;
public class TCPC
{
public static void main(String[] args) throws Exception
{
Socket sock=new Socket("127.0.01",4000);

System.out.println("Enter the filename");

BufferedReader keyRead=new BufferedReader(new InputStreamReader(System.in));

String fname=keyRead.readLine();

OutputStream ostream=sock.getOutputStream();

PrintWriter pwrite=new PrintWriter(ostream,true);

pwrite.println(fname);

InputStream istream=sock.getInputStream();

BufferedReader socketRead=new BufferedReader(new InputStreamReader(istream));

String str;
while((str=socketRead.readLine())!=null)
{
System.out.println(str);
```

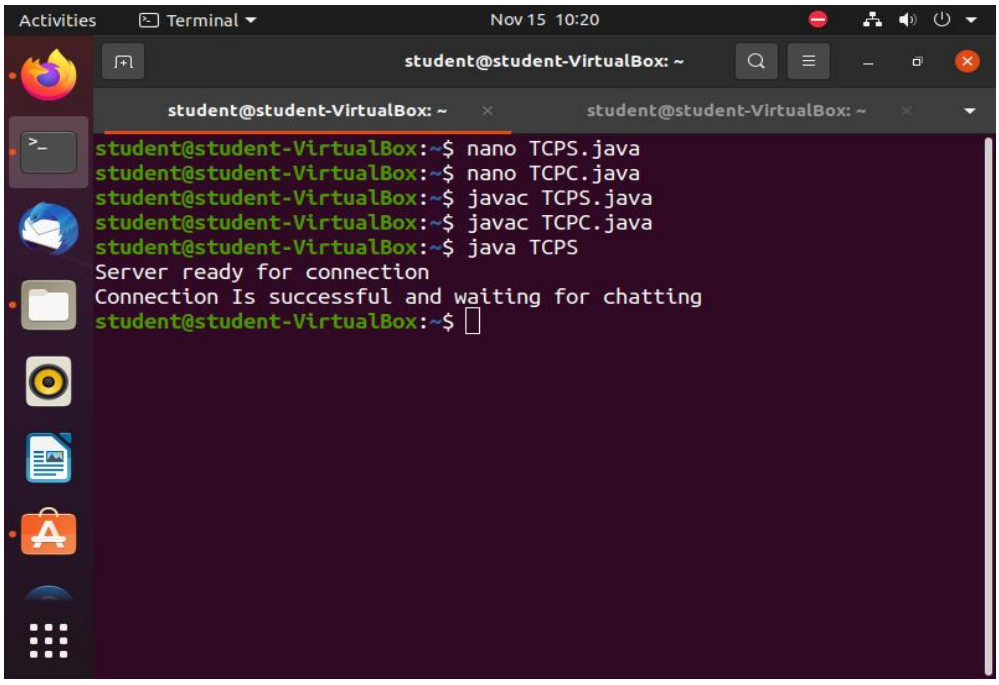
```
        }

        pwrite.close();
        socketRead.close();
        keyRead.close();
        }
}
```

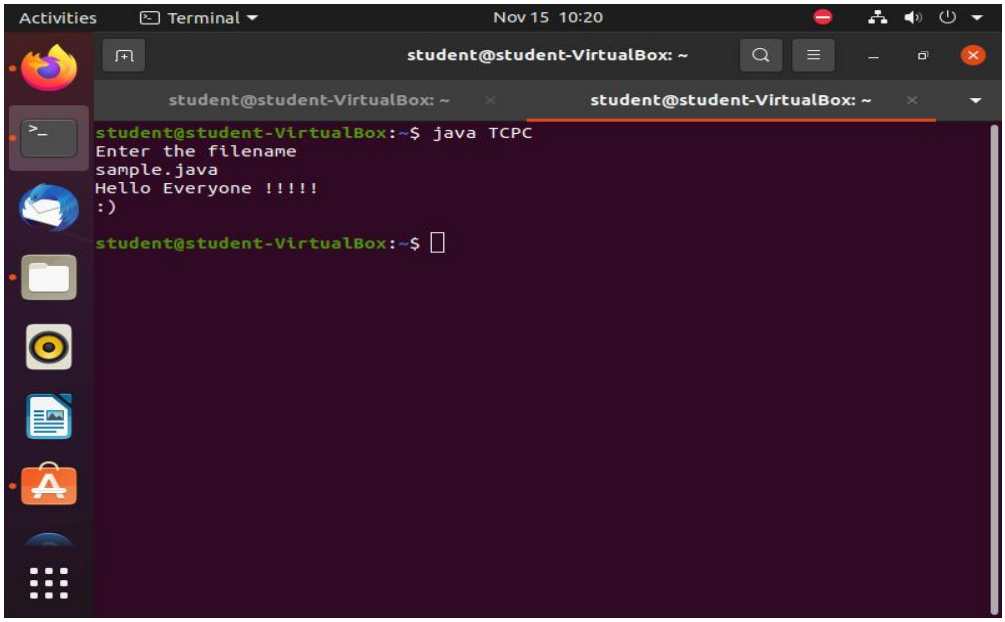
Press Ctrl + O (to write out).
Press Enter to confirm the file name.
Press Ctrl + X to exit nano.

```
javac TCPS.java
javac TCPC.java
java TCPS
java TCPC
```

```
ns lab1.tcl
```

OUTPUT




Activities
Terminal
Nov 15 10:20
student@student-VirtualBox: ~
student@student-VirtualBox: ~
student@student-VirtualBox: ~
student@student-VirtualBox:~$ java TCPC
Enter the filename
sample.java
Hello Everyone !!!!!
:)
student@student-VirtualBox:~$

7. Develop a program on a datagram socket for client/server to display the messages on client side, typed at the server side.

nano UDPServer.java

```
import java.net.*;
import java.net.InetAddress;
class UDPServer
{
public static void main(String args[])throws Exception
{
DatagramSocket serverSocket = new DatagramSocket(9876);
byte[] receiveData=new byte[1024];
byte[] sendData=new byte[1024];
while(true)
{
System.out.println("Server is Up");

DatagramPacket receivePacket=new DatagramPacket(receiveData, receiveData.length);
serverSocket.receive(receivePacket);

String sentence=new String(receivePacket.getData());
System.out.println("RECEIVED: " + sentence.trim()); // trim to remove padding spaces

InetAddress IPAddress=receivePacket.getAddress();
int port=receivePacket.getPort();

String capitalizedSentence=sentence.toUpperCase();
sendData=capitalizedSentence.getBytes();

DatagramPacket sendPacket=new DatagramPacket(sendData, sendData.length,
IPAddress, port);
serverSocket.send(sendPacket);
}
}
}
```

Press Ctrl + O (to write out).
Press Enter to confirm the file name.
Press Ctrl + X to exit nano.

nano UDPClient.java

```
import java.io.*;
import java.net.*;
import java.net.InetAddress;
class UDPClient
{
public static void main(String[] args)throws Exception
{
BufferedReader inFromUser=new BufferedReader(new InputStreamReader(System.in));

DatagramSocket clientSocket=new DatagramSocket();
InetAddress IPAddress=InetAddress.getByName("localhost");

byte[] sendData=new byte[1024];
byte[] receiveData=new byte[1024];

System.out.println("Enter the string to be converted into uppercase: ");
String sentence=inFromUser.readLine();

sendData=sentence.getBytes();

DatagramPacket sendPacket=new DatagramPacket(sendData, sendData.length,
IPAddress, 9876);
clientSocket.send(sendPacket);

DatagramPacket receivePacket=new DatagramPacket(receiveData, receiveData.length);
clientSocket.receive(receivePacket);

String modifiedSentence=new String(receivePacket.getData()).trim(); // trim to remove
padding spaces

System.out.println("FROM SERVER: " + modifiedSentence);

clientSocket.close();
}
}
```

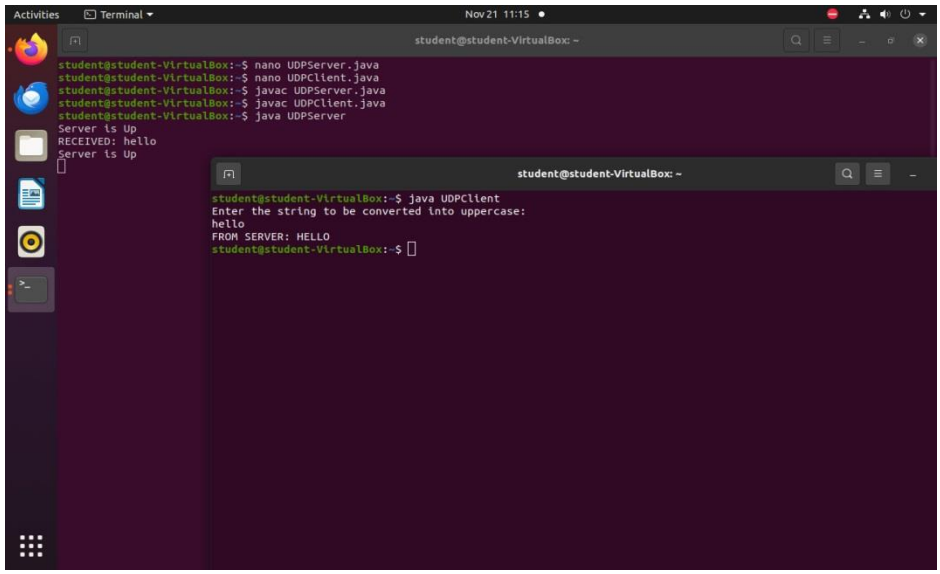
Press Ctrl + O (to write out).

Press Enter to confirm the file name.

Press Ctrl + X to exit nano.

javac UDPServer.java
javac UDPClient.java
java UDPServer
java UDPClient

OUTPUT



8. Develop a program for a simple RSA algorithm to encrypt and decrypt the data.

nano RSA.java

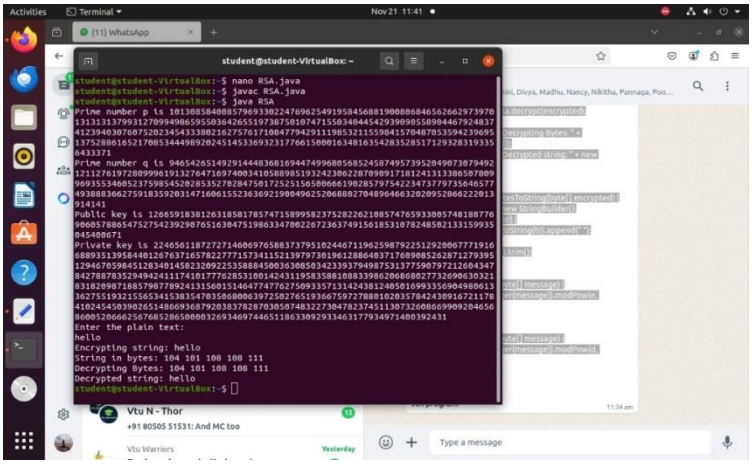
```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;
import java.math.BigInteger;
import java.util.Random;

public class RSA {
   private BigInteger p, q, N, phi, e, d;
   private int bitlength = 1024;
   private Random r;

   public RSA() {
      r = new Random();
      p = BigInteger.probablePrime(bitlength, r);
      q = BigInteger.probablePrime(bitlength, r);
      System.out.println("Prime number p is " + p);
      System.out.println("Prime number q is " + q);
      N = p.multiply(q);
      phi = p.subtract(BigInteger.ONE).multiply(q.subtract(BigInteger.ONE));
      e = BigInteger.probablePrime(bitlength / 2, r);
      while (phi.gcd(e).compareTo(BigInteger.ONE) > 0 && e.compareTo(phi) < 0) {
         e.add(BigInteger.ONE);
      }
      System.out.println("Public key is " + e);
      d = e.modInverse(phi);
      System.out.println("Private key is " + d);
   }

   public RSA(BigInteger e, BigInteger d, BigInteger N) {
      this.e = e;
      this.d = d;
      this.N = N;
   }
```

```
    public static void main(String[] args) throws IOException {
        RSA rsa = new RSA();

        // Use BufferedReader instead of DataInputStream
        BufferedReader in = new BufferedReader(new InputStreamReader(System.in));

        System.out.println("Enter the plain text:");
        String testString = in.readLine();  // Read the input from the user

        System.out.println("Encrypting string: " + testString);
        System.out.println("String in bytes: " + bytesToString(testString.getBytes()));

        byte[] encrypted = rsa.encrypt(testString.getBytes());
        byte[] decrypted = rsa.decrypt(encrypted);

        System.out.println("Decrypting Bytes: " + bytesToString(decrypted));
        System.out.println("Decrypted string: " + new String(decrypted));
    }

    private static String bytesToString(byte[] encrypted) {
        StringBuilder test = new StringBuilder();
        for (byte b : encrypted) {
            test.append(Byte.toString(b)).append(" ");
        }
        return test.toString().trim();
    }

    public byte[] encrypt(byte[] message) {
        return (new BigInteger(message)).modPow(e, N).toByteArray();
    }

    public byte[] decrypt(byte[] message) {
        return (new BigInteger(message)).modPow(d, N).toByteArray();
    }
}
```

Press Ctrl + O (to write out).
Press Enter to confirm the file name.
Press Ctrl + X to exit nano.

```
javac RSA.java
```

java RSA


Activities
Terminal
Nov 21 11:41
(11) WhatsApp
student@student-VirtualBox: ~
student@student-VirtualBox:~$ nano RSA.java
student@student-VirtualBox:~$ javac RSA.java
student@student-VirtualBox:~$ java RSA
Prime number p is 10130858400857969330224769625491958456881900808684656266297397013113133179931270994986595503646265519738750107471550340445429398905589804467924837412394037607520234543133002162757617108477942911985321155984157048705359423969513752886165217085344498920245145336932317766150001634816354283528517129328319335643337
Prime number q is 9465426514929144483681694474996805685245874957395204907307949212112761972809996191327647169740034105880985193242306228709091718124131338650780996935534605237598545202853527028475017252515650066619028579754223473779735646577493888366275918359203147160615523636921908496252068802704896466320209528662220139141141
Public key is 12665918381263185817857471589958237528226210857476593300574818877690605788654752754239290765163047519863347002267236374915618531078248502133159935045408671
Private key is 22465611872727146069765883737951024467119625987922512920067771916688935139584401267637165782277715734115213979730196128864037176090852628712793951294670598451283401458232092255388845003630805034233937949875313775907972126043474427887835294942411174101777628531001424311958358810883398620686802773269030321831828987185579877892411156015144477477627509335711424381240581699356904980613362755193215565341538354703506800639725027651936675972788010203578424309167211784102454503902651486693687920383782870305074832273047823745113073260866990920465660085266625076852865800032693469744651186330929334631779349714003924311
Enter the plain text:
hello
Encrypting string: hello
String in bytes: 104 101 108 108 111
Decrypting Bytes: 104 101 108 108 111
Decrypted string: hello
student@student-VirtualBox:~$
Vtu N - Thor
+91 80505 51531: And MC too
Vtu Warriors
Yesterday
Type a message

9. Develop a program for congestion control using a leaky bucket algorithm.

nano lab7.java

```
import java.util.Scanner;
import java.lang.*;
public class lab7 {
public static void main(String[] args)
{
int i;
int a[]=new int[20];
int buck_rem=0,buck_cap=4,rate=3,sent,recv;
Scanner in = new Scanner(System.in);
System.out.println("Enter the number of packets");
int n = in.nextInt();
System.out.println("Enter the packets");
for(i=1;i<=n;i++)
a[i]= in.nextInt();
System.out.println("Clock \t packet size \t accept \t sent \t remaining");
for(i=1;i<=n;i++)
{
if(a[i]!=0)
{
if(buck_rem+a[i]>buck_cap)
recv=-1;
else
{
recv=a[i];
buck_rem+=a[i];
}
}
else
recv=0;
if(buck_rem!=0)
{
if(buck_rem<rate)
{sent=buck_rem;
buck_rem=0;
}
else
{
```

```
        sent=rate;
        buck_rem=buck_rem-rate;
        }
        }
        else
        sent=0;
        if(recv==-1)
        System.out.println(+i+ "\t\t" +a[i]+ "\t dropped \t" +  sent +"\t" +buck_rem);
        else
        System.out.println(+i+ "\t\t" +a[i] +"\t\t" +recv +"\t" +sent + "\t" +buck_rem);
        }
        }
}
```

Press Ctrl + O (to write out).
Press Enter to confirm the file name.
Press Ctrl + X to exit nano.

```
javac lab7.java
java lab7
```