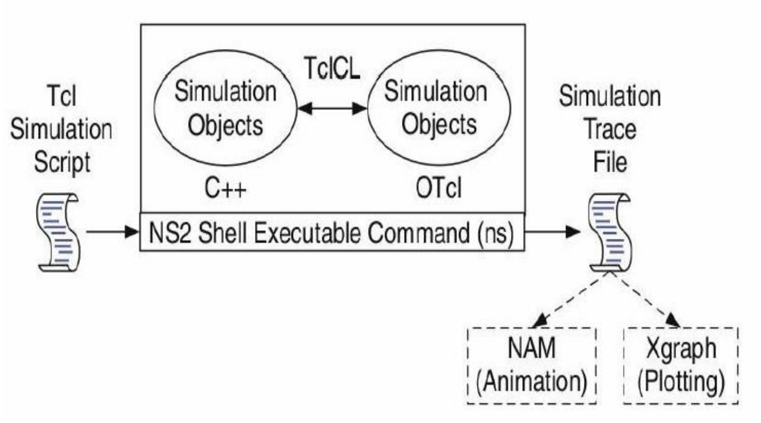
SIMULATION USING NS-2

Introduction to NS-2:

NS2 is an open-source simulation tool that runs on Linux. It is a discreet event simulator targeted at networking research and provides substantial support for simulation of routing, multicast protocols and IP protocols, such as UDP, TCP, RTP and SRM over wired and wireless (local and satellite) networks.

- Widely known as NS2, is simply an event driven simulation tool.
- Useful in studying the dynamic nature of communication networks.
- Simulation of wired as well as wireless network functions and protocols (e.g., routing algorithms, TCP, UDP) can be done using NS2.
- In general, NS2 provides users with a way of specifying such network protocols and simulating their corresponding behaviors.

Basic Architecture of NS2



Tcl scripting

• Tcl is a general purpose scripting language.[Interpreter]

• Tcl runs on most of the platforms such as Unix, Windows, and Mac.

• The strength of Tcl is its simplicity.

• It is not necessary to declare a data type for variable prior to the usage.

Wired TCL Script Components

- Create the event scheduler
- Open new files & turn on the tracing
- Create the nodes
- Setup the links
- Configure the traffic type (e.g., TCP, UDP,etc)
- Set the time of traffic generation (e.g., CBR,FTP)
- Terminate the simulation

NS Simulator Preliminaries.

1. Initialization and termination aspects of the ns simulator.

2. Definition of network nodes, links, queues and topology.

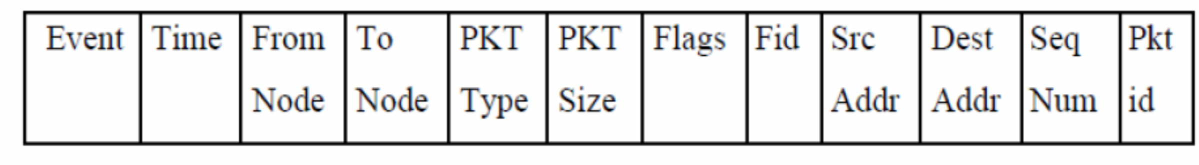
3. Definition of agents and of applications.

4. The nam visualization tool.

5. Tracing and random variables

Structure of Trace Files

When tracing into an output ASCII file, the trace is organized in12 fields as follows in fig shown below:



Event	Time	From Node	To Node	PKT Type	PKT Size	Flags	Fid	Src Addr	Dest Addr	Seq Num	Pkt id

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.

```
# This script is created by NSG2 beta1
# <http://wushoupong.googlepages.com/nsg>
#===================================
# Simulation parameters setup
#===================================
set val(stop) 10.0 ;# time of simulation end
#===================================
# Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]
#Open the NS trace file
set tracefile [open 1.tr w]
$ns trace-all $tracefile
#Open the NAM trace file
set namfile [open 1.nam w]
$ns namtrace-all $namfile
#===================================
# Nodes Definition
#===================================
#Create 3 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
#===================================
# Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n1 $n0 100.0Mb 10ms DropTail
$ns queue-limit $n1 $n0 50
$ns duplex-link $n0 $n2 100.0Mb 10ms DropTail
$ns queue-limit $n0 $n2 50
#Give node position (for NAM)
$ns duplex-link-op $n1 $n0 orient right-up
$ns duplex-link-op $n0 $n2 orient right-down
#===================================
# Agents Definition
#===================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
```

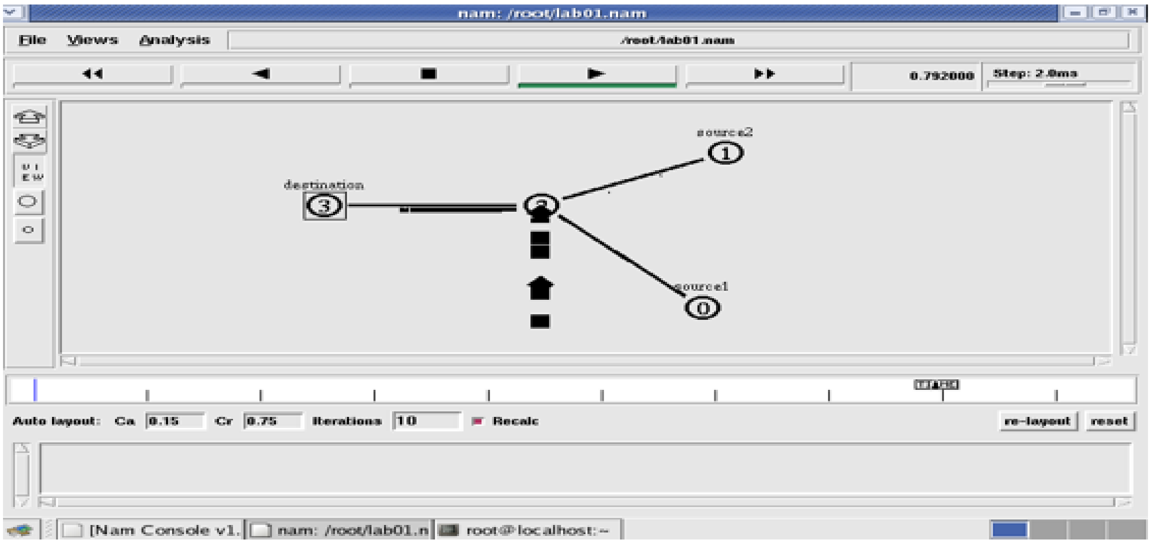
```
$ns attach-agent $n1 $tcp0
set sink1 [new Agent/TCPSink]
$ns attach-agent $n2 $sink1
$ns connect $tcp0 $sink1
$tcp0 set packetSize_ 1500

#===================================
# Applications Definition
#===================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
$ns at 0.1 "$ftp0 start"
$ns at 2.0 "$ftp0 stop"

#===================================
# Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
global ns tracefile namfile
$ns flush-trace
close $tracefile
close $namfile
exec nam 1.nam &
exit 0
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
$ns at $val(stop) "finish"
$ns at $val(stop) "puts \"done\" ; $ns halt"
$ns run
```

AWK file:

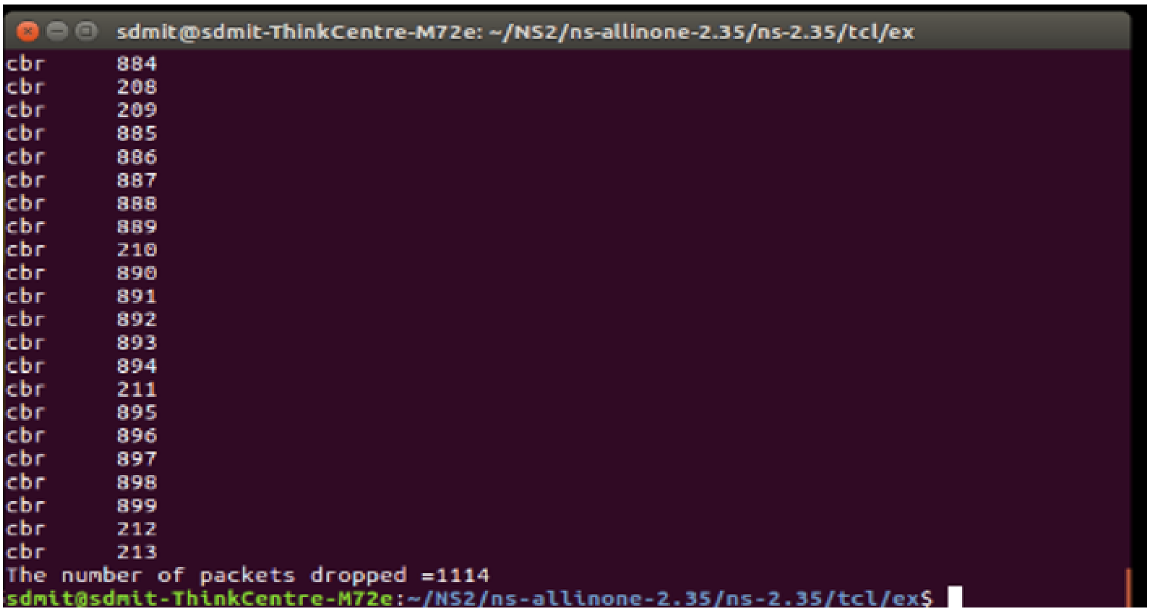
```
BEGIN {
c=0;
}
{
if($1=="d")
                {
                        c++;
                        printf("%s\t%s\n",$5,$11);
                }
}
```

```
END {
printf("The number of packets dropped =%d\n",c);
}
```

Topology:



Output:



```
sdmit@sdmit-ThinkCentre-M72e: ~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex
cbr      884
cbr      208
cbr      209
cbr      885
cbr      886
cbr      887
cbr      888
cbr      889
cbr      210
cbr      890
cbr      891
cbr      892
cbr      893
cbr      894
cbr      211
cbr      895
cbr      896
cbr      897
cbr      898
cbr      899
cbr      212
cbr      213
The number of packets dropped =1114
sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$
```

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.

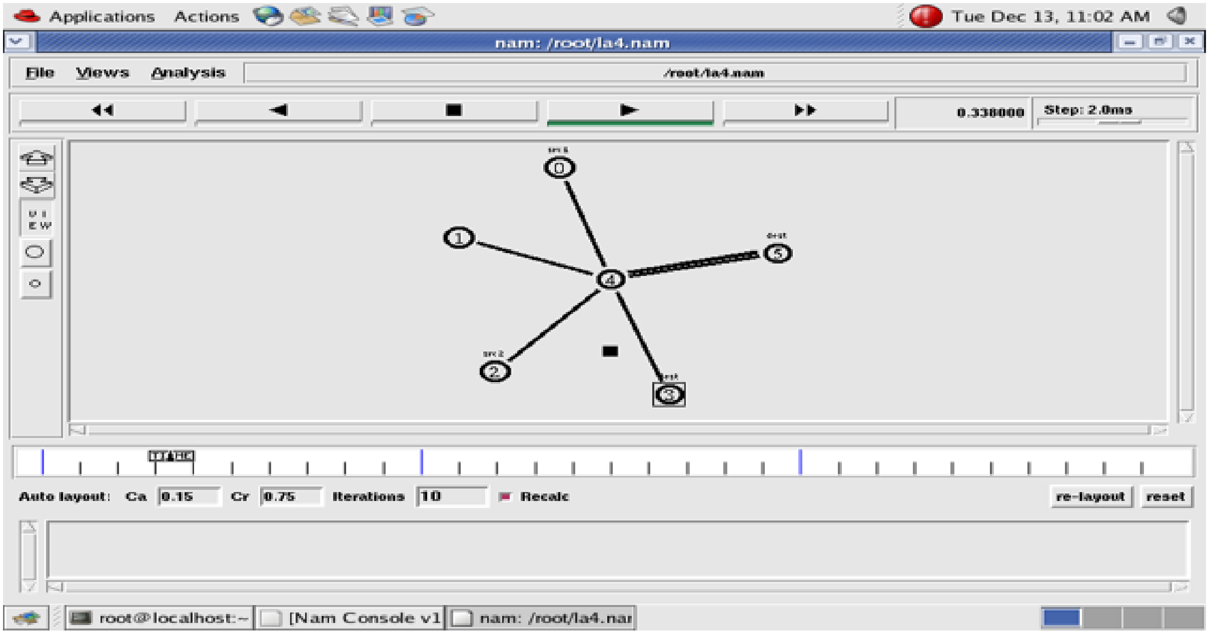
```
set ns [new Simulator]
set nf [open prog2.nam w]
$ns namtrace-all $nf
set nd [open prog2.tr w]
$ns trace-all $nd

proc finish { } {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam prog2.nam &
exit 0
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
set n6 [$ns node]

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

$ns duplex-link $n4 $n0 1Mb 10ms DropTail
$ns duplex-link $n5 $n0 1Mb 10ms DropTail
$ns duplex-link $n6 $n0 1Mb 10ms DropTail

Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts "node [$node_ id] received ping answer from \ $from with round-trip-time $rtt ms."
}

set p1 [new Agent/Ping]
set p2 [new Agent/Ping]
set p3 [new Agent/Ping]
set p4 [new Agent/Ping]
set p5 [new Agent/Ping]
set p6 [new Agent/Ping]

$ns attach-agent $n1 $p1
$ns attach-agent $n2 $p2
$ns attach-agent $n3 $p3
$ns attach-agent $n4 $p4
```

```
$ns attach-agent $n5 $p5
$ns attach-agent $n6 $p6

$ns queue-limit $n0 $n4 3
$ns queue-limit $n0 $n5 2
$ns queue-limit $n0 $n6 2

$ns connect $p1 $p4

$ns connect $p2 $p5
$ns connect $p3 $p6

$ns at 0.2 "$p1 send"
$ns at 0.4 "$p2 send"
$ns at 0.6 "$p3 send"
$ns at 1.0 "$p4 send"
$ns at 1.2 "$p5 send"
$ns at 1.4 "$p6 send"
$ns at 2.0 "finish"
$ns run
```

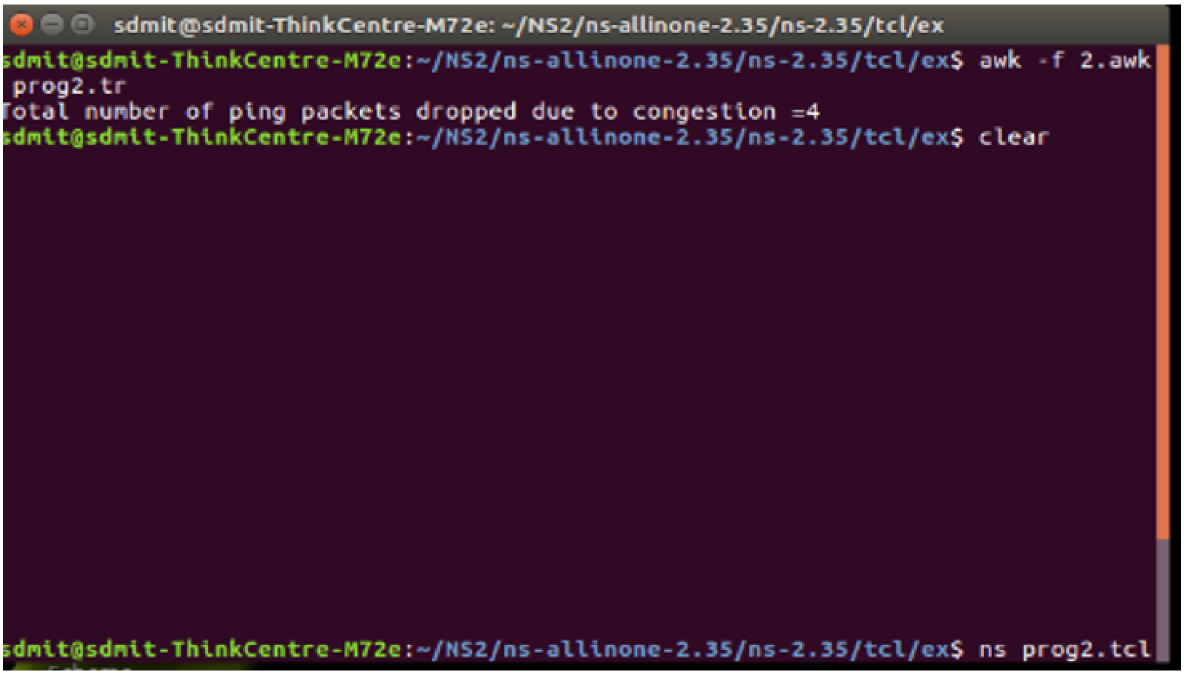
AWK file:

```
BEGIN {
drop=0;
}
{
if($1=="d")
{
drop++;
}
}
END {
printf("Total number of %s packets dropped due to congestion =%d\n",$5,drop);
}
```

Topology:



Output:



```
sdmit@sdmit-ThinkCentre-M72e: ~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex
sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$ awk -f 2.awk
 prog2.tr
Total number of ping packets dropped due to congestion =4
sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$ clear















sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$ ns prog2.tcl
```

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

AWK:

```
BEGIN{
}
{
```

```
if($6=="cwnd_")
printf("%f\t%f\t\n",$1,$7);
}
END{
}
```

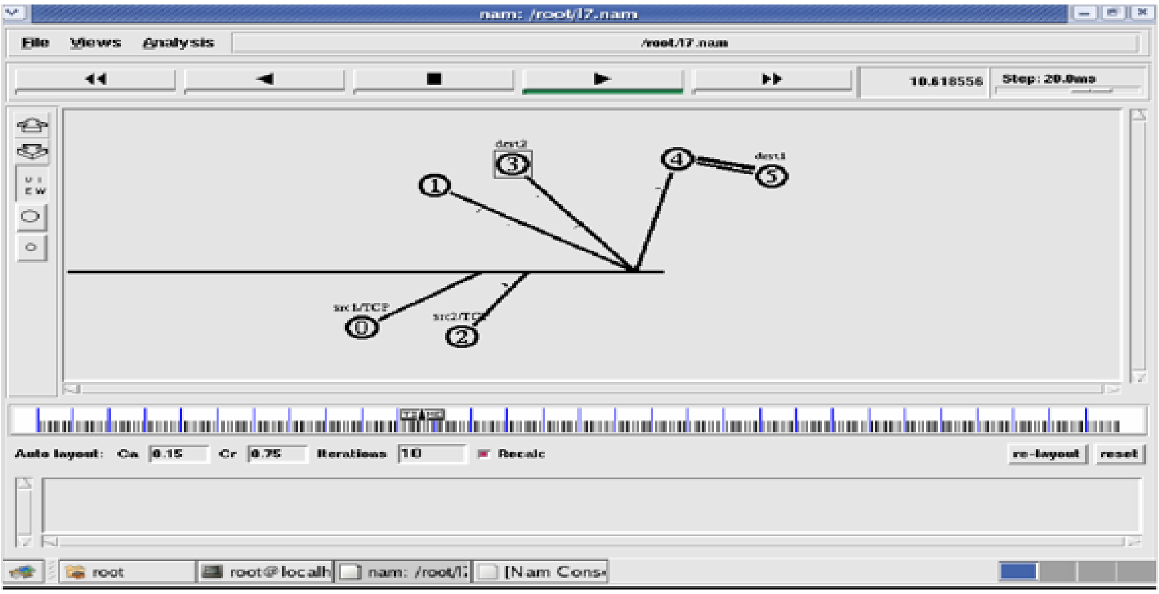
TCL:

```
# This script is created by NSG2 beta1
# <http://wushoupong.googlepages.com/nsg>

#===================================
#     Simulation parameters setup
#===================================
set val(stop)   10.0                         ;# time of simulation end

#===================================
#        Initialization
#===================================
#Create a ns simulator
set ns [new Simulator]

#Open the NS trace file
set tracefile [open aa3.tr w]
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open aa3.nam w]
$ns namtrace-all $namfile

#===================================
#        Nodes Definition
#===================================
#Create 6 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]
set n5 [$ns node]
$ns make-lan "$n0 $n1 $n2 $n3 $n4" 100Mb 100ms LL Queue/DropTail Mac/802_3
#===================================
#        Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n4 $n5 100.0Mb 10ms DropTail
$ns queue-limit $n4 $n5 50

#Give node position (for NAM)
$ns duplex-link-op $n4 $n5 orient right-up
```

```
#===================================
#        Agents Definition
#===================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set sink2 [new Agent/TCPSink]
$ns attach-agent $n3 $sink2
$ns connect $tcp0 $sink2
$tcp0 set packetSize_ 1500

#Setup a TCP connection
set tcp1 [new Agent/TCP]
$ns attach-agent $n1 $tcp1
set sink3 [new Agent/TCPSink]
$ns attach-agent $n5 $sink3
$ns connect $tcp1 $sink3
$tcp1 set packetSize_ 1500


#===================================
#        Applications Definition
#===================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp1
$ns at 0.1 "$ftp0 start"
$ns at 16.0 "$ftp0 stop"

#Setup a FTP Application over TCP connection
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp0
$ns at 0.1 "$ftp1 start"
$ns at 16.0 "$ftp1 stop"

set file1 [open file1.tr w]
$tcp1 attach $file1
set file2 [open file2.tr w]
$tcp0 attach $file2
$tcp1 trace cwnd_
$tcp0 trace cwnd_
#===================================
#        Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
   global ns tracefile namfile
   $ns flush-trace
   close $tracefile
   close $namfile
   exec nam aa3.nam &
```

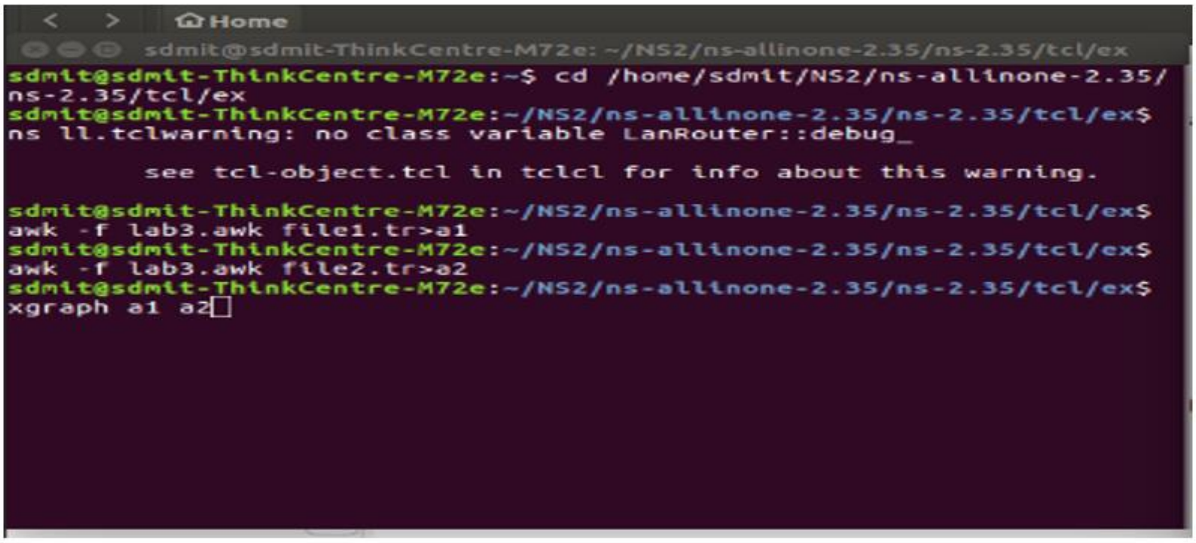
```
    exit 0
}

$ns at 0.1 "$ftp0 start"
$ns at 5 "$ftp0 stop"
$ns at 7 "$ftp0 start"
$ns at 0.2 "$ftp1 start"
$ns at 8 "$ftp1 stop"
$ns at 14 "$ftp0 stop"
$ns at 10 "$ftp1 start"
$ns at 15 "$ftp1 stop"
$ns at 16 "finish"
$ns run
```

Topology:

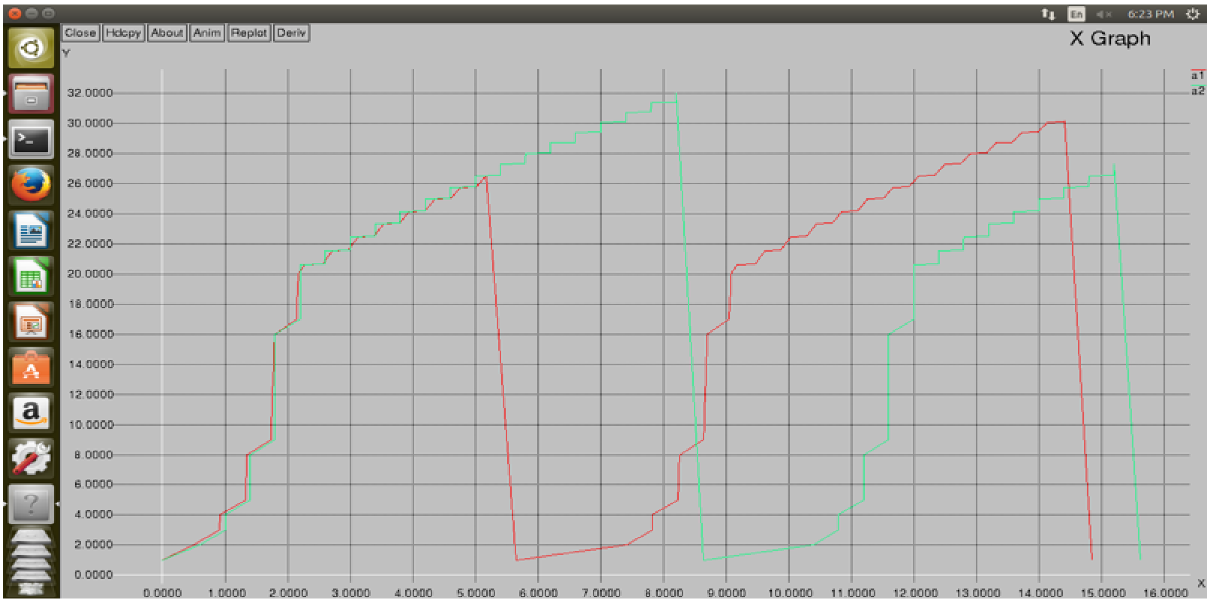


Auto layout: Ca 0.15 Cr 0.75 Iterations 10 Recalc

Output:


Home
sdmit@sdmit-ThinkCentre-M72e: ~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex

```
sdmit@sdmit-ThinkCentre-M72e:~$ cd /home/sdmit/NS2/ns-allinone-2.35/
ns-2.35/tcl/ex
sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$
ns ll.tclwarning: no class variable LanRouter::debug_

        see tcl-object.tcl in tclcl for info about this warning.

sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$
awk -f lab3.awk file1.tr>a1
sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$
awk -f lab3.awk file2.tr>a2
sdmit@sdmit-ThinkCentre-M72e:~/NS2/ns-allinone-2.35/ns-2.35/tcl/ex$
xgraph a1 a2
```



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

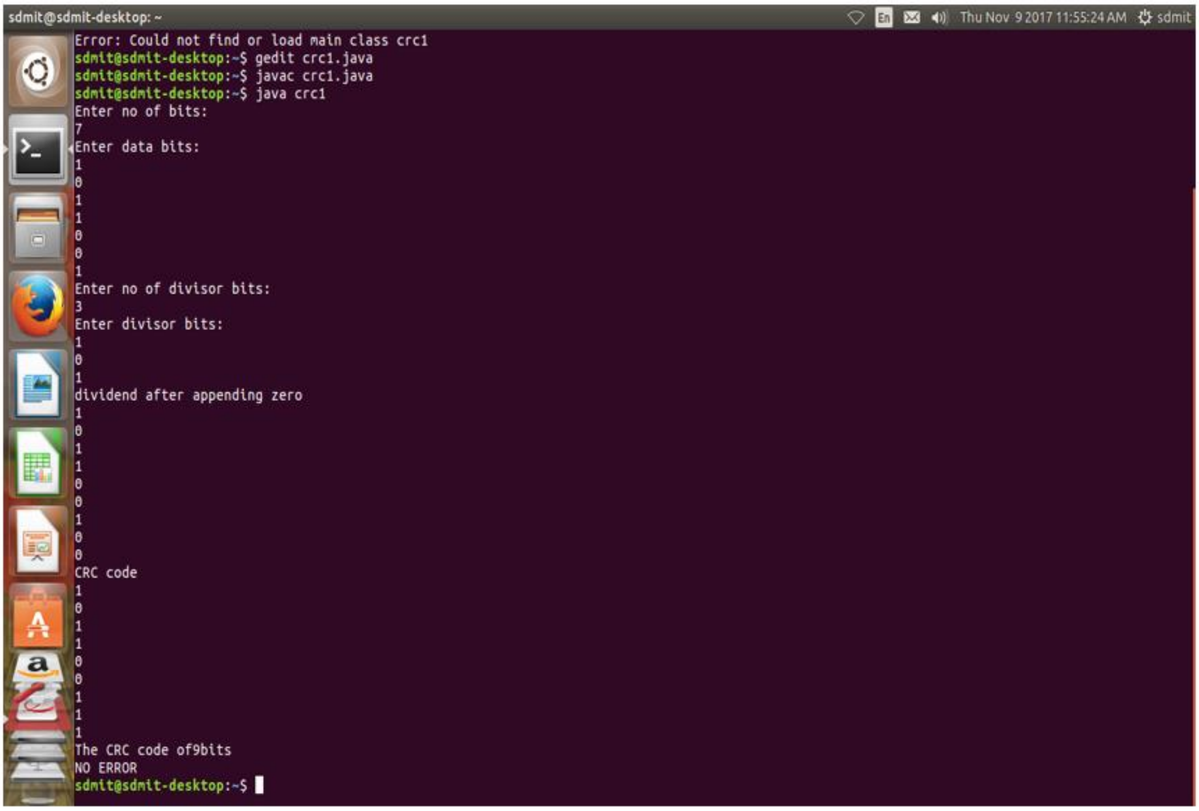
Source code:

```
import java.io.*;
import java.util.*;
class crc1{
public static void main(String args[]){
Scanner s=new Scanner(System.in);
System.out.println("Enter no of bits:");
int n=s.nextInt();
int data[];
data=new int[n];
System.out.println("Enter data bits:");
for(int i=0;i<n;i++)
data[i]=s.nextInt();
```

```
System.out.println("Enter no of divisor bits:");
int m=s.nextInt();
int divisor[];
divisor=new int[m];
System.out.println("Enter divisor bits:");
for(int j=0;j<m;j++)
divisor[j]=s.nextInt();
int len=n+m-1;
int div[]=new int[len];
int rem[]=new int[len];
int crc[]=new int[len];

int src[]=new int[len];
for(int i=0;i<data.length;i++)
div[i]=data[i];
System.out.println("dividend after appending zero");
for(int i=0;i<div.length;i++)
System.out.println(div[i]);
for(int j=0;j<div.length;j++)
rem[j]=div[j];
rem=divide(div,divisor,rem);
for(int i=0;i<div.length;i++)
crc[i]=div[i]^rem[i];
System.out.println("CRC code");
for(int i=0;i<crc.length;i++)
System.out.println(crc[i]);
System.out.println("The CRC code of"+len+"bits");
for(int i=0;i<src.length;i++)
for(int j=0;j<crc.length;j++)
rem[j]=crc[j];
rem=divide(crc,divisor,rem);
for(int i=0;i<rem.length;i++)
{
if(rem[i]!=0)
{
System.out.println("ERROR!!!!!");
break;
}

if(i==rem.length-1)
System.out.println("NO ERROR");
}
}
static int[] divide(int div[],int divisor[],int rem[]){
int cur=0;
while(true)
{
```

```
for(int i=0;i<divisor.length;i++)
rem[cur+i]=(rem[cur+i]^divisor[i]);
while(rem[cur]==0 && cur!=rem.length-1)
cur++;
if((rem.length-cur)<(divisor.length))
break;
}
return rem;
}
}
```

Output:



```
Error: Could not find or load main class crc1
sdmit@sdmit-desktop:~$ gedit crc1.java
sdmit@sdmit-desktop:~$ javac crc1.java
sdmit@sdmit-desktop:~$ java crc1
Enter no of bits:
7
Enter data bits:
1
0
1
1
0
0
1
Enter no of divisor bits:
3
Enter divisor bits:
1
0
1
dividend after appending zero
1
0
1
1
0
0
1
0
0
CRC code
1
0
1
1
0
0
1
1
1
The CRC code of9bits
NO ERROR
sdmit@sdmit-desktop:~$ 
```

5. Develop a program to implement a sliding window protocol in the data link layer.

Source code:

```
import java.util.Scanner;
class SlidingWindowProtocol38{
public static void main(String[] args){
Scanner scanner=new Scanner(System.in);
System.out.println("Enter the total number of frames to send: ");
int totalFrames=scanner.nextInt();
System.out.println("Enter the window size: ");
int windowSize=scanner.nextInt();
int sent=0;
int ack=0;
while(sent<totalFrames){
System.out.println("Sending frames in the window: ");
for(int i=0;i<windowSize&&sent+i<totalFrames;i++){
System.out.println("Sent frame: "+(sent+i));
}
System.out.println("Enter the no. of frames acknowledged (0 to resebd all): ");
int acknowledged=scanner.nextInt();
if(acknowledged==0){
System.out.println("No frames acknowledged. Resending all frames in the window...");
}else{
ack+=acknowledged;
sent=ack;
System.out.println("Ackonwledged upto frames: "+(ack-1));
System.out.println("Sliding window...");
}
if(ack>=totalFrames){
System.out.println("All frames successfully transmitted!");
break;
}
}
scanner.close();
}
}
```

Output:

```
Enter the total number of frames to send:
5
Enter the window size:
2
Sending frames in the window:
Sent frame: 0
Sent frame: 1
Enter the number of frames acknowledged (0 to resend all);
1
Acknowledged up to frames:0
Sliding window…
Sending frames in the window:
Sent frame: 1
Sent frame: 2
Enter the number of frames acknowledged (0 to resend all);
2
Acknowledged up to frames:2
```

```
Sliding window…
Sending frames in the window:
Sent frame: 3
Sent frame: 4
Enter the number of frames acknowledged (0 to resend all);
2
Acknowledged up to frames:4
Sliding window…
All frames are successfully transmitted!!!
```

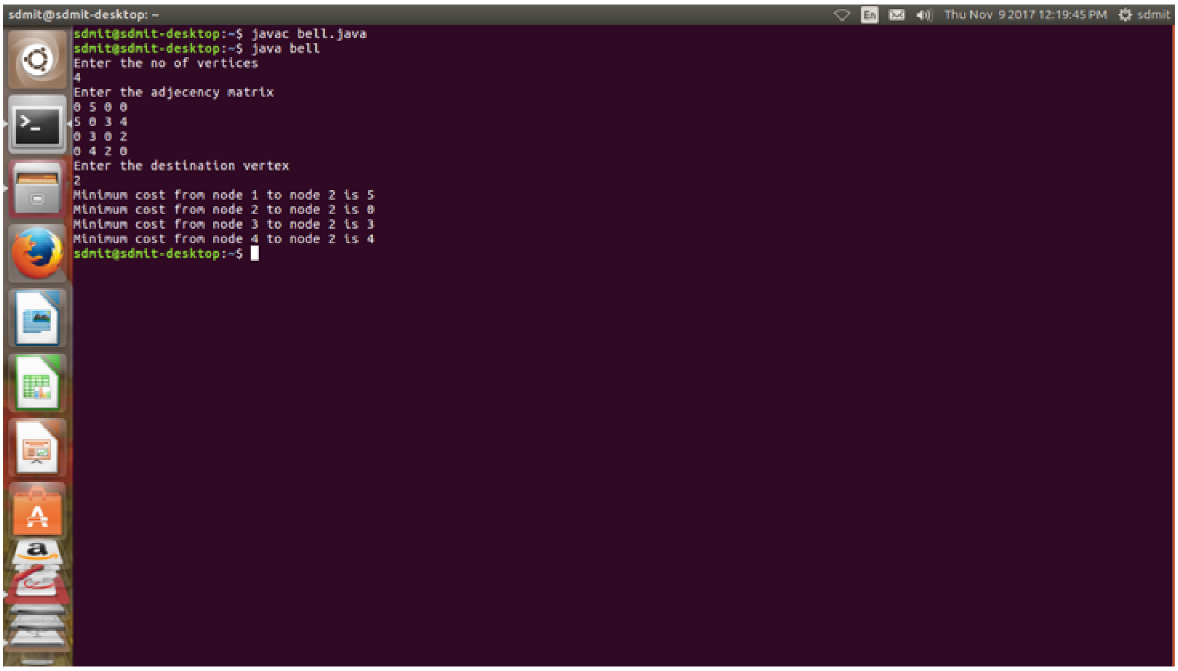
6. Develop a program to find the shortest path between vertices using the Bellman-Ford and path vector routing algorithm.

Source Code:

```
import java.util.*;
public class bell
{
    private int d[];
    private int n;
    public static final int MAX=999;
    public bell(int n)
    {
        this.n=n;
        d=new int[n+1];
    }
    public void belleval(int source,int a[][])
    {
        for(int node=1;node<=n;node++)
        {
            d[node]=MAX;
        }
    d[source]=0;
    for(int node=1;node<=n-1;node++)
    {
        for(int sn=1;sn<=n;sn++)
        {
            for(int dn=1;dn<=n;dn++)
            {
                if(a[sn][dn]!=MAX)
                {
                    if(d[dn]>d[sn]+a[sn][dn])
                    d[dn]=d[sn]+a[sn][dn];
}
            }
        }
    }
    for(int sn=1;sn<=n;sn++)
        {
            for(int dn=1;dn<=n;dn++)
```

```
                        {
                                if(a[sn][dn]!=MAX)
                                {
                                        if(d[dn]>d[sn]+a[sn][dn])
                                        System.out.println("The negetive edge cycle");
                                }
                        }
                }
        for(int v=1;v<=n;v++)
        {
        System.out.println("Minimum cost from node "+v+" to node "+source+" is "+d[v]);
        }
  }
public static void main(String args[])
        {
                int n=0;
                int source;
                System.out.println("Enter the no of vertices");
                Scanner s=new Scanner(System.in);
                n=s.nextInt();
                int a[][]=new int[n+1][n+1];
                System.out.println("Enter the adjecency matrix");
for(int i=1;i<=n;i++)
                {
                        for(int j=1;j<=n;j++)
                        {
                                a[i][j]=s.nextInt();
                                if(i==j)
                                {
                                        a[i][j]=0;
                                        continue;
                                }
                                if(a[i][j]==0)
                                {
                                        a[i][j]=MAX;
                                }
                        }
                }
        System.out.println("Enter the destination vertex");
                source=s.nextInt();
                bell b=new bell(n);
                b.belleval(source,a);
        }
}
```

Output:



```
sdmit@sdmit-desktop:~$ javac bell.java
sdmit@sdmit-desktop:~$ java bell
Enter the no of vertices
4
Enter the adjecency matrix
0 5 0 0
5 0 3 4
0 3 0 2
0 4 2 0
Enter the destination vertex
2
Minimum cost from node 1 to node 2 is 5
Minimum cost from node 2 to node 2 is 0
Minimum cost from node 3 to node 2 is 3
Minimum cost from node 4 to node 2 is 4
sdmit@sdmit-desktop:~$
```

7. 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.

Client :

Source Code:

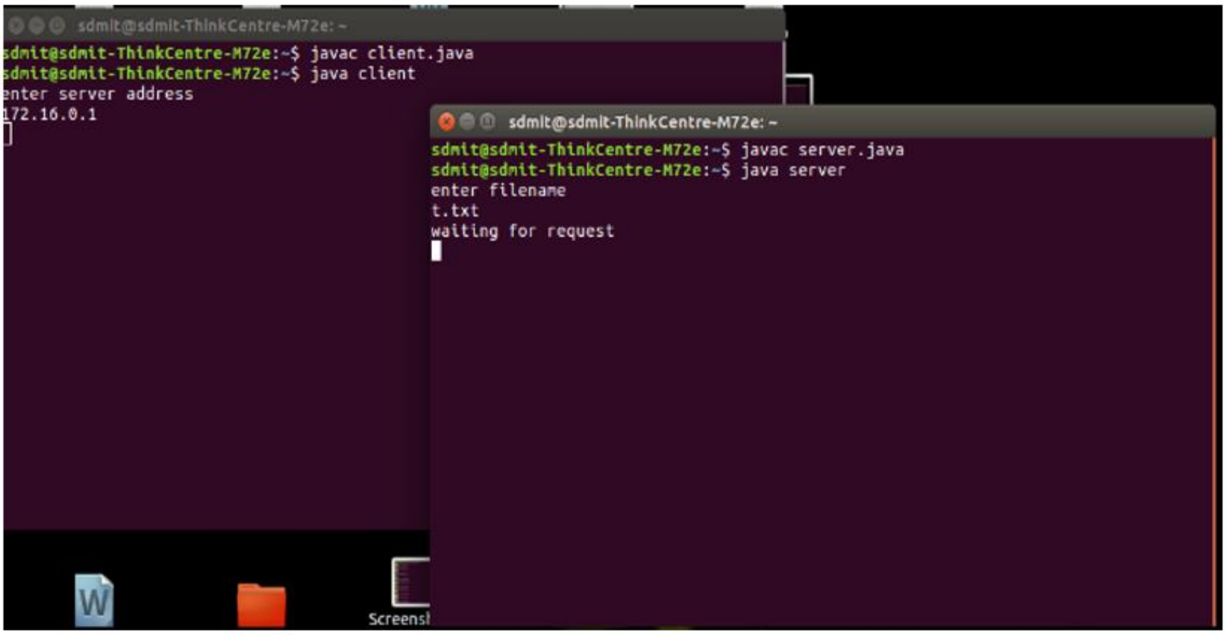
```
import java.io.*;
import java.net.Socket;
import java.util.Scanner;
class client
{
public static void main(String args[])throws Exception
{
String address=" ";
Scanner sc=new Scanner(System.in);
System.out.println("enter server address");
address=sc.nextLine();
Socket s=new Socket(address,5000);
DataInputStream din=new DataInputStream(s.getInputStream());
DataOutputStream dout=new DataOutputStream(s.getOutputStream());
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Send Get to Start...");
String str=" ",filename=" ";
try
{
while(!str.equals("start"))
str=br.readLine();
dout.writeUTF(str);
dout.flush();
filename=din.readUTF();
System.out.println("Recieving file:"+filename);
filename="client"+filename;
System.out.println("Saving as file:"+filename);
long sz=Long.parseLong(din.readUTF());
System.out.println("File size:"+(sz/(1024*1024))+"MB");
byte b[]=new byte[1024];
System.out.println("Recieving file....");
FileOutputStream fos=new FileOutputStream(new File(filename),true);
long bytesRead;
do
{
bytesRead=din.read(b,0,b.length);
fos.write(b,0,b.length);
}
while(!(bytesRead<1024));
System.out.println("completed");
fos.close();
dout.close();
s.close();
}                    catch(EOFException e)
}
}
```

Server:

<u>Source code:</u>

```
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.*;
class server
{
public static void main(String args[])throws Exception
{
String filename;
System.out.println("enter filename");
Scanner sc=new Scanner(System.in);
filename=sc.nextLine();
sc.close();
while(true)
{
ServerSocket ss=new ServerSocket(5000);
System.out.println("waiting for request");
Socket s=ss.accept();
System.out.println("connected with "+s.getInetAddress().toString());
DataInputStream din=new DataInputStream(s.getInputStream());
DataOutputStream dout=new DataOutputStream(s.getOutputStream());
try
{
String str=" ";
str=din.readUTF();
System.out.println("Send Get .....ok");
if(!str.equals("stop"))
{
System.out.println("Sending File:"+filename);
dout.writeUTF(filename);
dout.flush();
File f=new File(filename);
FileInputStream fin=new FileInputStream(f);
long sz=(int)f.length();
byte b[]=new byte[1024];
int read;
dout.writeUTF(Long.toString(sz));
dout.flush();
System.out.println("size:"+sz);
System.out.println("Buf size:"+ss.getReceiveBufferSize());
while((read=fin.read(b))!=-1)
{
dout.write(b,0,read);
dout.flush();
}
fin.close();
System.out.println("...ok");
dout.flush();
}
dout.writeUTF("stop");
System.out.println("Send complete");
dout.flush();
}
catch(Exception e)
{
e.printStackTrace();
System.out.println("An error occured");
}
din.close();
s.close();
ss.close();
}
}
}
```

Output:



sdmit@sdmit-ThinkCentre-M72e: ~
sdmit@sdmit-ThinkCentre-M72e: ~
Screensl

```
sdmit@sdmit-ThinkCentre-M72e:~$ javac client.java
sdmit@sdmit-ThinkCentre-M72e:~$ java client
enter server address
172.16.0.1
```

```
sdmit@sdmit-ThinkCentre-M72e:~$ javac server.java
sdmit@sdmit-ThinkCentre-M72e:~$ java server
enter filename
t.txt
waiting for request
```

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

Source Code:

Client:

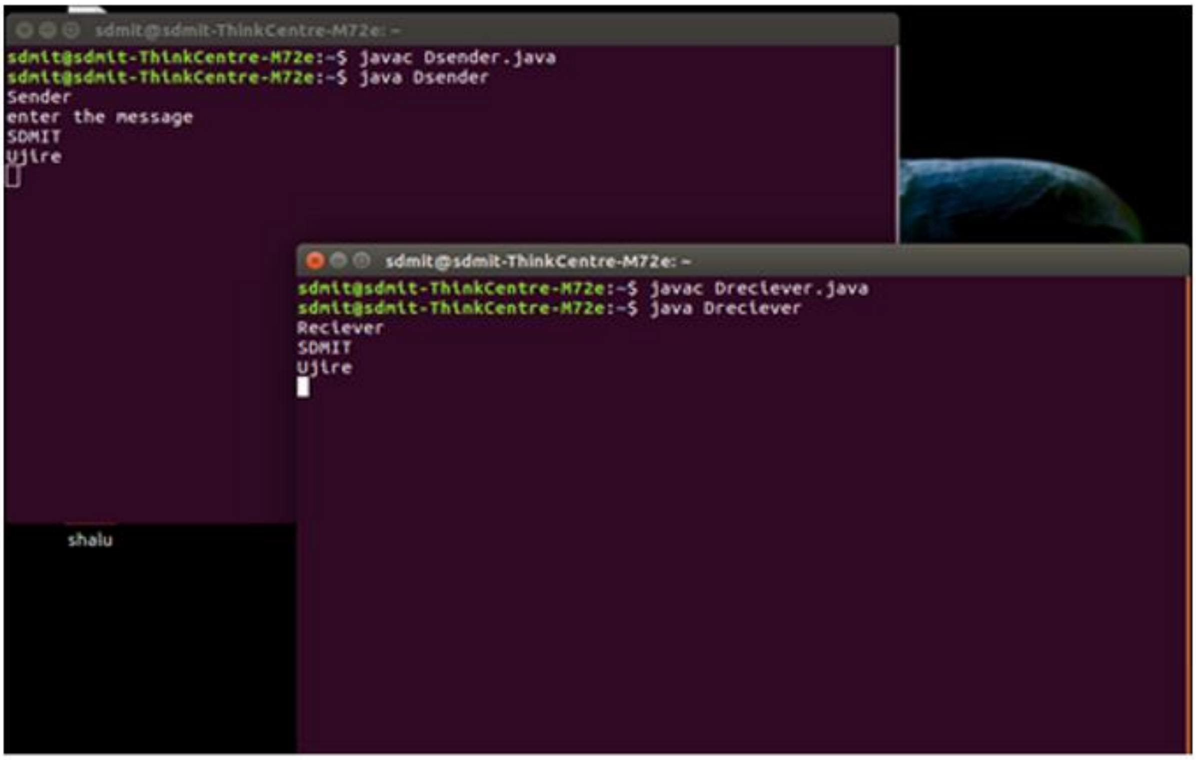
```
import java.net.*;
import java.util.Scanner;
public class Dsender
{

public static void main(String args[])throws Exception
{
System.out.println(&quot;Sender&quot;);
DatagramSocket ds=new DatagramSocket();
Scanner scanner=new Scanner(System.in);
System.out.println(&quot;enter the message&quot;);
while(true)
{
String msg=scanner.nextLine();
InetAddress ip=InetAddress.getByName(&quot;127.0.0.1&quot;);
DatagramPacket dp=new DatagramPacket(msg.getBytes(),msg.length(),ip,3000);

ds.send(dp);
}
}

}
```

Server:

```
import java.util.Scanner;
import java.net.*;
public class Dreciever
{
```

```
public static void main(String[] args)throws Exception
{
byte[]buf=new byte[1024];
System.out.println(&quot;Reciever&quot;);
DatagramSocket ds=new DatagramSocket(3000);
while(true)
{
DatagramPacket dp=new DatagramPacket(buf,1024);
ds.receive(dp);
String msg=new String(dp.getData(),0,dp.getLength());
System.out.println(msg);
}
}

}
```

Output:


sdmit@sdmit-ThinkCentre-M72e: ~
sdmit@sdmit-ThinkCentre-M72e: ~
shalu

```
sdmit@sdmit-ThinkCentre-M72e:~$ javac Dsender.java
sdmit@sdmit-ThinkCentre-M72e:~$ java Dsender
Sender
enter the message
SDMIT
Ujire
```

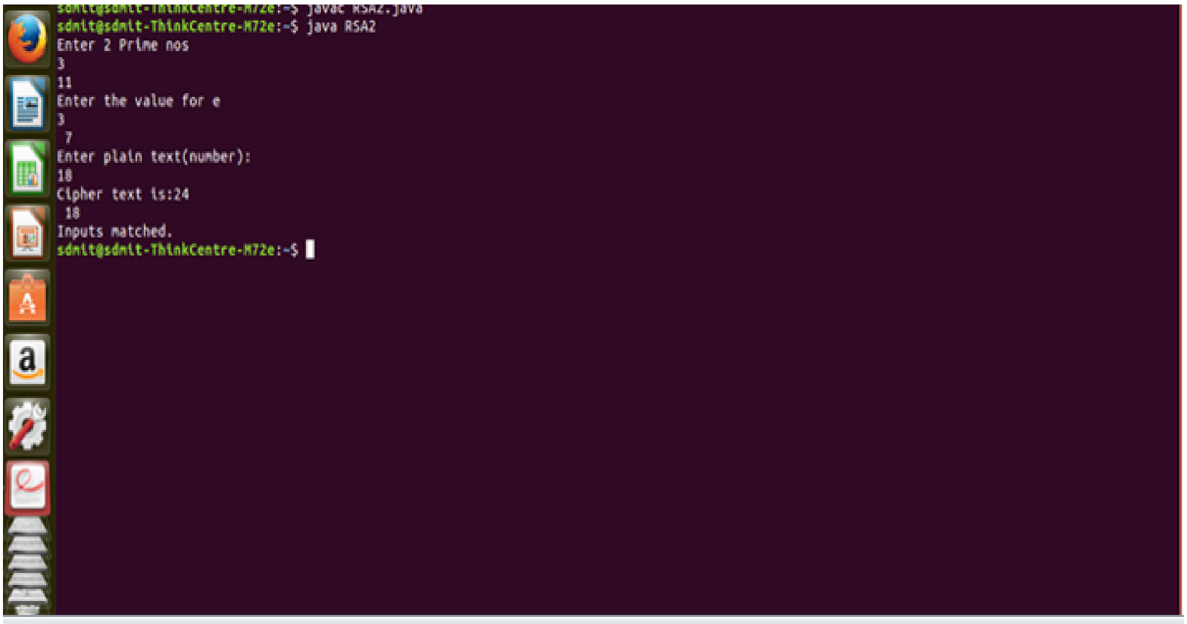
```
sdmit@sdmit-ThinkCentre-M72e:~$ javac Dreciever.java
sdmit@sdmit-ThinkCentre-M72e:~$ java Dreciever
Reciever
SDMIT
Ujire
```

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

Source Code:

```
import java.util.*;
import java.math.*;
import java.util.Random;
import java.util.*;
class RSA2
{
public static void main(String args[])
{
Scanner sc=new Scanner(System.in);
System.out.println(&quot;enter two prime numbers&quot;);

int p=sc.nextInt();
int q=sc.nextInt();
int n=p*q;
int fi=(p-1)*(q-1);
System.out.println(&quot;enter the value for e&quot;);
int e=sc.nextInt();
while(true)
{
if(findGCD(fi,e)!=1)
{
System.out.println(&quot;enter a valid value for e&quot;);
e=sc.nextInt();
}
else
break;
}
int d=findD(e,fi);
System.out.println(&quot;d value is &quot;+d);
System.out.println(&quot;enter plain text(number):&quot;);
int m=sc.nextInt();
int c=(int)(Math.pow(m,e)%n);
System.out.println(&quot;cipher text is:&quot; +c);
int x=(int)(Math.pow(c,d)%n);
System.out.println(&quot; &quot;+x);
if(x==m)
System.out.println(&quot;input matched&quot;);
else
System.out.println(&quot;input mismatched&quot;);
}

static int findD(int e,int fi)
{
int b;
for(int i=1;i&lt;=fi;i++)
{
b=(i*e)%fi;
if(b==1)
return i;
}
return 0;
}
static int findGCD(int number1,int number2)
{
if(number2==0)
{
return number1;
} return findGCD(number2,number1%number2);
} }
```

Output:



```
sdmit@sdmit-ThinkCentre-M72e:~$ javac RSA2.java
sdmit@sdmit-ThinkCentre-M72e:~$ java RSA2
Enter 2 Prime nos
3
11
Enter the value for e
3
 7
Enter plain text(number):
18
Cipher text is:24
 18
Inputs matched.
sdmit@sdmit-ThinkCentre-M72e:~$
```

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

Source code:

```
import java.util.*;
public class LBA{
public static void main(String args[])
{
int op;
Scanner s = new Scanner(System.in);
int remain = 0;
int bktCap = 0;
System.out.println(&quot;Enter the total capacity of the bucket: &quot;);
bktCap = s.nextInt();
System.out.println(&quot;Enter the number of inputs(Value of n): &quot;);
int n = s.nextInt();
int pkt[] = new int[10];
System.out.println(&quot;Enter the output datarate: &quot;);
int dr = s.nextInt();
System.out.println(&quot;\nEnter the input values: &quot;);
for(int i=0;i&lt;n;i++)
{
int val = s.nextInt();
pkt[i] = val;
}
for(int i=0;i&lt;=n;i++)
{

int total = pkt[i]+remain;
if(total&gt;bktCap)
{
if(pkt[i]&gt;bktCap)
{
System.out.println(&quot;Incoming packet size &quot;+pkt[i]+&quot; bytes is greater
than bucket
capacity of &quot;+bktCap+&quot; bytes&quot;);
System.out.println(&quot;Bucket size Exceeded- Packet is REJECTED\n&quot;);

}
}
else
{

remain += pkt[i];
System.out.println(&quot;Incoming packet size is: &quot;+pkt[i]);

if(remain&lt;=dr)
{
op = remain;
remain = 0;
}
else
{
op = dr;
remain -= dr;
}
System.out.println(&quot;Packet of size &quot;+op+&quot; is transmitted&quot;);
System.out.println(&quot;Remaining capacity is: &quot;+remain);
System.out.println();

}
}
}
}
```

Output:

```
Enter the total capacity of the bucket: 50
Enter the number of inputs (Value of n): 4
Enter the output data rate: 20

Enter the input values: 20, 20, 60, 30.
Incoming packet size is: 20

Packet of size 20 is transmitted
```

```
Remaining capacity is: 0

Incoming packet size is: 20
Packet of size 20 is transmitted
Remaining capacity is: 0

Incoming packet size 60 bytes is greater than bucket capacity of 50 bytes
Bucket size Exceeded- Packet is REJECTED

Incoming packet size is: 30
Packet of size 20 is transmitted
Remaining capacity is: 10

Incoming packet size is: 0
Packet of size 10 is transmitted
Remaining capacity is: 0
```