PART- A (NS2 SIMULATION PROGRAMS)

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.

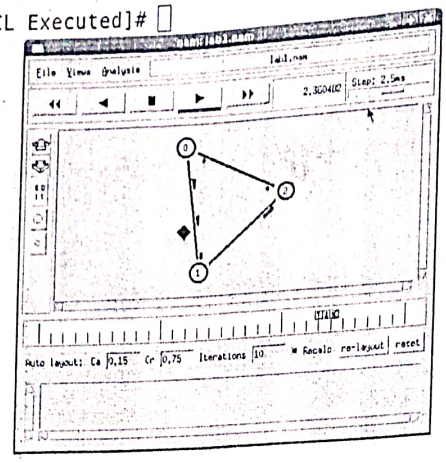
```
set ns [new Simulator]
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set f [open lab1.tr w]
$ns trace-all $f
set nf [open lab1.nam w]
$ns namtrace-all $nf
$ns duplex-link $n0 $n2 5Mb 2ms DropTail
$ns duplex-link $n1 $n2 5Mb 2ms DropTail
$ns duplex-link $n0 $n1 1.5Mb 10ms DropTail
$ns queue-limit $n0 $n2 10
$ns queue-limit $n0 $n2 10
$ns queue-limit $n0 $n2 10
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
set udp1 [new Agent/UDP]
$ns attach-agent $n2 $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
set null0 [new Agent/Null]
$ns attach-agent $n1 $null0
set null1 [new Agent/Null]
$ns attach-agent $n1 $null1
$ns connect $udp0 $null0
$ns connect $udp1 $null1
$ns at 1.0 "$cbr0 start"
$ns at 1.1 "$cbr1 start"
set tcp [new Agent/TCP]
set sink [new Agent/TCPSink]
$ns attach-agent $n0 $tcp
$ns attach-agent $n2 $sink
$ns connect $tcp $sink
set ftp [new Application/FTP]
$ftp attach-agent $tcp
$ns at 1.2 "$ftp start "
$ns at 3.0 "finish"
```

```
proc finish {} {
global ns f nf
$ns flush-trace
close $f
close $nf
puts "running nam..."
exec nam lab1.nam &
exit 0
}
$ns run
```

AWK file (lab1.awk)

```
#AWK file: Use geditor Note: The symbol # refer comments in NS2 TCL Code.
#immediately after BEGIN should open braces ,,{,,

BEGIN{ c=0;}
{
  if($1=="d")
  {    c++;
       printf("%s\t%s\n",$5,$11);
  }
}
END{ printf("The number of packets dropped =%d\n",c); }


  #Run the simulation program
   #            [root@localhost~]# ns lab1.tcl
   # After simulation is completed run awk file to see the output ,
    #           [root@localhost~]# awk –f lab1.awk lab1.tr
   # To see the trace file contents open the file as ,
     #          [root@localhost~]# vi lab1.tr
```

```
Applications Places System
File Edit View Terminal Tabs Help
[root@localhost TCL Executed]# ns  lab1.tcl
running nam...
[root@localhost TCL Executed]#
```




```
root@localhost:~/TCL...   nam: lab1.nam   [Nam Console v1.14]

[root@localhost TCL Executed]# ns lab1.tcl
running nam...
[root@localhost TCL Executed]# ls
5wdone.tcl  6wdone.tcl  ex1.nam  lab1.awk  lab1.nam  lab1.tcl  lab1.tcl~  lab1.tr  lab2.nam  lab2.tcl  lab3.awk  lab3.nam  lab3.tcl  lab4.awk
[root@localhost TCL Executed]# gedit lab1.awk
[root@localhost TCL Executed]# awk -f lab1.awk lab1.tr
tcp     24
tcp     26
tcp     28
tcp     30
tcp     32
tcp     34
tcp     36
tcp     38
tcp     51
tcp     52
tcp     149
tcp     247
tcp     345
tcp     443
tcp     541
tcp     639
tcp     737
tcp     835
tcp     933
tcp     1031
The number of packets dropped =20
```

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 lab2.nam w]
$ns namtrace-all $nf
set tf [open lab2.tr w]
$ns trace-all $tf

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

$n4 shape box

$ns duplex-link $n0 $n4 1005mb 1ms DropTail
$ns duplex-link $n1 $n4  50mb 1ms DropTail
$ns duplex-link $n2 $n4 2000mb 1ms DropTail
$ns duplex-link $n3 $n4  200mb 1ms DropTail
$ns duplex-link $n4 $n5 1mb 1ms DropTail

set p1 [new Agent/Ping]
$ns attach-agent $n0 $p1
$p1 set packetSize_ 50000
$p1 set interval_ 0.0001

set p2 [new Agent/Ping]
$ns attach-agent $n1 $p2

set p3 [new Agent/Ping]
$ns attach-agent $n2 $p3
$p3 set packetSize_ 30000
$p3 set interval_ 0.00001

set p4 [new Agent/Ping]
$ns attach-agent $n3 $p4
$p4 set interval_ 0.00001

set p5 [new Agent/Ping]
$ns attach-agent $n5 $p5
```

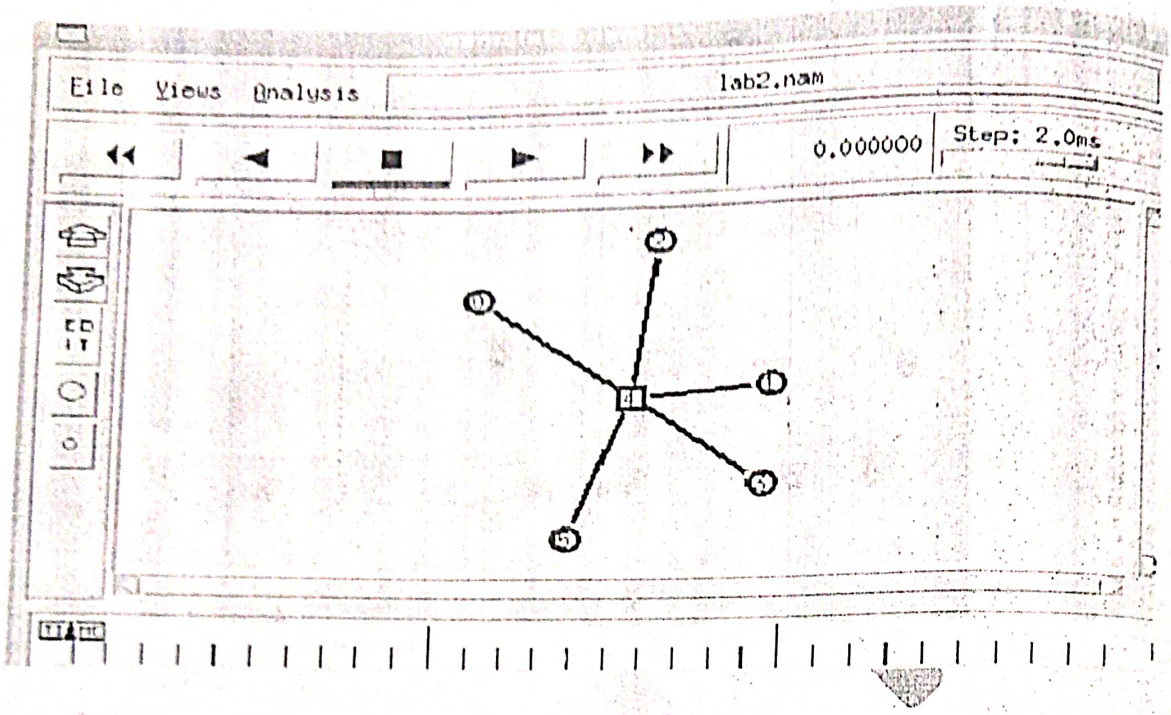
```
$ns queue-limit $n0 $n4 5
$ns queue-limit $n2 $n4 3
$ns queue-limit $n4 $n5 2

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

$ns connect $p1 $p5
$ns connect $p3 $p4
proc finish { } {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam lab2.nam &
exit 0
}

$ns at 0.1 "$p1 send"
$ns at 0.2 "$p1 send"
$ns at 0.3 "$p1 send"
$ns at 0.4 "$p1 send"
$ns at 0.5 "$p1 send"
$ns at 0.6 "$p1 send"
$ns at 0.7 "$p1 send"
$ns at 0.8 "$p1 send"
$ns at 0.9 "$p1 send"
$ns at 1.0 "$p1 send"
$ns at 1.1 "$p1 send"
$ns at 1.2 "$p1 send"
$ns at 1.3 "$p1 send"
$ns at 1.4 "$p1 send"
$ns at 1.5 "$p1 send"
$ns at 1.6 "$p1 send"
$ns at 1.7 "$p1 send"
$ns at 1.8 "$p1 send"
$ns at 1.9 "$p1 send"
$ns at 2.0 "$p1 send"
$ns at 2.1 "$p1 send"
$ns at 2.2 "$p1 send"
$ns at 2.3 "$p1 send"
$ns at 2.4 "$p1 send"
$ns at 2.5 "$p1 send"
$ns at 2.6 "$p1 send"
$ns at 2.7 "$p1 send"
$ns at 2.8 "$p1 send"
$ns at 2.9 "$p1 send"
$ns at 0.1 "$p3 send"
$ns at 0.2 "$p3 send"
```

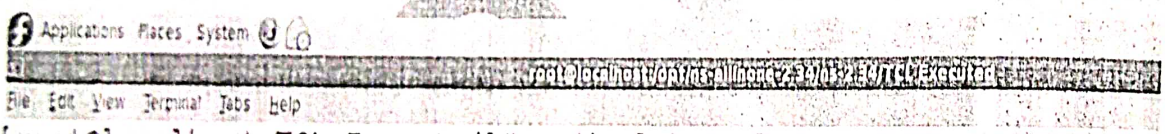
```
$ns at 0.3 "$p3 send"
$ns at 0.4 "$p3 send"
$ns at 0.5 "$p3 send"
$ns at 0.6 "$p3 send"
$ns at 0.7 "$p3 send"
$ns at 0.8 "$p3 send"
$ns at 0.9 "$p3 send"
$ns at 1.0 "$p3 send"
$ns at 1.1 "$p3 send"
$ns at 1.2 "$p3 send"
$ns at 1.3 "$p3 send"
$ns at 1.4 "$p3 send"
$ns at 1.5 "$p3 send"
$ns at 1.6 "$p3 send"
$ns at 1.7 "$p3 send"
$ns at 1.8 "$p3 send"
$ns at 1.9 "$p3 send"
$ns at 2.0 "$p3 send"
$ns at 2.1 "$p3 send"
$ns at 2.2 "$p3 send"
$ns at 2.3 "$p3 send"
$ns at 2.4 "$p3 send"
$ns at 2.5 "$p3 send"
$ns at 2.6 "$p3 send"
$ns at 2.7 "$p3 send"
$ns at 2.8 "$p3 send"
$ns at 2.9 "$p3 send"
$ns at 3.0 "finish"
$ns run
```

AWK FILE (lab2.awk)

```
BEGIN{
drop=0;

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

}






```
[root@localhost TCL Executed]# gedit lab2.tcl
[root@localhost TCL Executed]# gedit lab2.awk
[root@localhost TCL Executed]# ls
5wdone.tcl  lab1.awk   lab1.tcl   lab2.nam  lab3.awk  lab4.awk   pgr
5wdone.tcl  lab1.awk~  lab1.tcl~  lab2.tcl  lab3.nam  lab4.awk~
ex1.nam     lab1.nam   lab1.tr    lab2.tr   lab3.tcl  lab4.tcl
[root@localhost TCL Executed]# gedit lab2.awk
[root@localhost TCL Executed]# awk -f lab2.awk lab2.tr
Total number of ping packets dropped due to congestion =20
[root@localhost TCL Executed]#
```

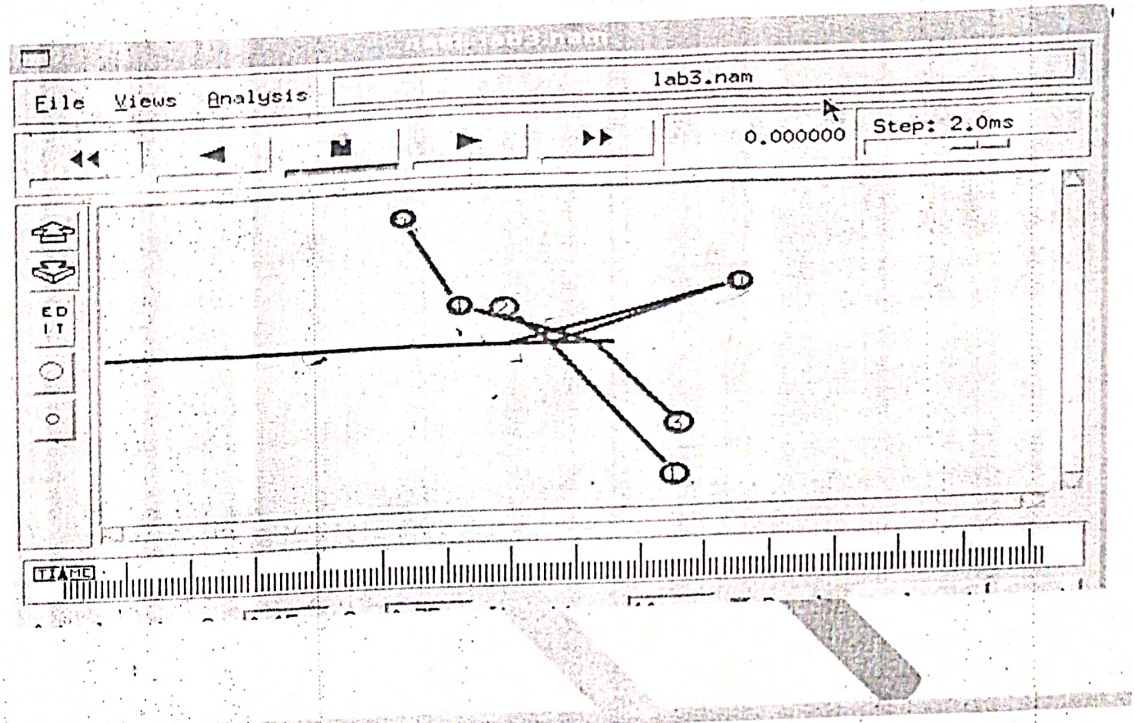
3.Implement an Ethernet LAN using n nodes and set multiple traffic nodes and plot congestion window for different source / destination.

```
set ns [new Simulator]

 set tf [open lab3.tr w]
 $ns trace-all $tf
 set nf [open lab3.nam w]
 $ns namtrace-all $nf

set n0 [$ns node]
$n0 color "magenta"
$n0 label "src1"

set n1 [$ns node]
set n2 [$ns node]
 $n2 color "magenta"
 $n2 label "src2"

  set n3 [$ns  node]
  $n3 color "blue"
  $n3 label "dest2"

set n4 [$ns node]
set n5 [$ns node]
$n5 color "blue"
$n5 label "dest1"

$ns make-lan "$n0 $n1 $n2 $n3 $n4" 100Mb 100ms LL Queue/DropTail Mac/802_3

$ns duplex-link $n4 $n5 1Mb 1ms DropTail

 set tcp0 [new Agent/TCP]
 $ns attach-agent $n0 $tcp0
 set ftp0 [new Application/FTP]
 $ftp0 attach-agent $tcp0
 $ftp0 set packetSize_ 500
 $ftp0 set interval_ 0.0001

 set sink5 [new Agent/TCPSink]
 $ns attach-agent $n5 $sink5
 $ns connect $tcp0 $sink5
 set tcp2 [new Agent/TCP]
 $ns attach-agent $n2 $tcp2

 set ftp2 [new Application/FTP]
```

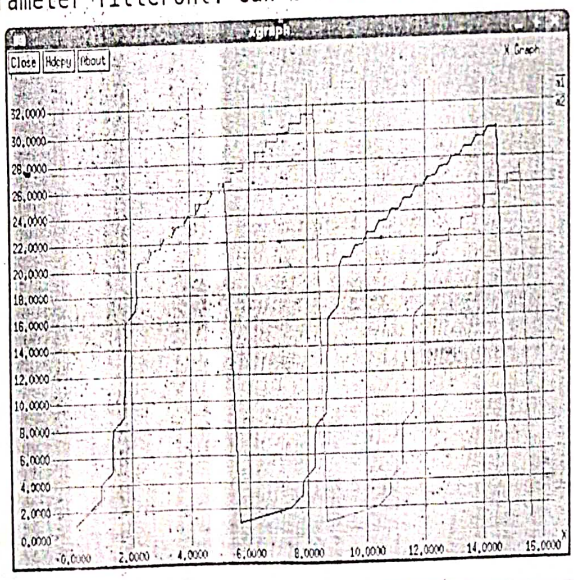
```
Sftp2 attach-agent Stcp2
Sftp2 set packetSize_ 600
Sftp2 set interval_ 0.001

set sink3 [new Agent/TCPSink]
Sns attach-agent $n3 $sink3
Sns connect $tcp2 $sink3

set file1 [open file1.tr w]
$tcp0 attach $file1
set file2 [open file2.tr w]
$tcp2 attach $file2
$tcp0 trace cwnd_
$tcp2 trace cwnd_

proc finish {} {
global ns nf tf
$ns flush-trace
close $tf
close $nf
exec nam lab3.nam &
exit 0
}
$ns at 0.1 "$ftp0 start"
$ns at 5 "$ftp0 stop"
$ns at 7 "$ftp0 start"
$ns at 0.2 "$ftp2 start"
$ns at 8 "$ftp2 stop"
$ns at 14 "$ftp0 stop"
$ns at 10 "$ftp2 start"
$ns at 15 "$ftp2 stop"
$ns at 16 "finish"
$ns run
```

AWK FILE (lab3.awk)

```
#cwnd:- means congestion window

BEGIN{
}
{
if($6=="cwnd_") /* don't leave space after writing cwnd_ */ printf("%f\t%f\t\n",$1,$7);
/* you must put \n in printf */
}
END{
}
```




```
File Edit View Terminal Tabs Help
[root@localhost TCL Executed]# awk -f lab3.awk file1.tr > a1
[root@localhost TCL Executed]# awk -f lab3.awk file2.tr > a2
[root@localhost TCL Executed]# xgraph a1 a2
Parameter LabelFont: can't translate `helvetica-10' into a font (defaulting to `fixed')
Parameter TitleFont: can't translate `helvetica-18' into a font (defaulting to `fixed')
]
```



4. Implement simple ESS and with transmitting nodes in wire-less LAN by simulation and determine the performance with respect to transmission of packets.

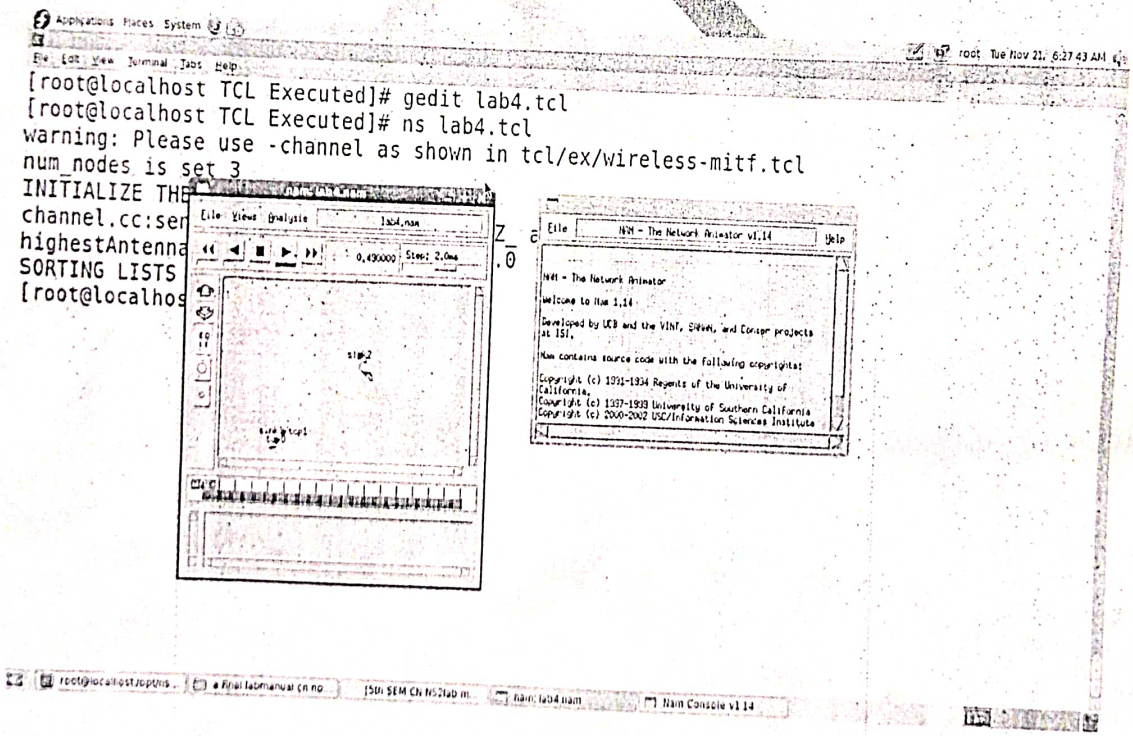
```
set ns [new Simulator]
set tf [open lab4.tr w]
$ns trace-all $tf
set topo [new Topography]
$topo load_flatgrid 1000 1000
set nf [open lab4.nam w]
$ns namtrace-all-wireless $nf 1000 1000
$ns node-config -adhocRouting DSDV \
                -llType LL \
                -macType Mac/802_11 \
                -ifqType Queue/DropTail \
                -ifqLen 50 \
                -phyType Phy/WirelessPhy \
                -channelType Channel/WirelessChannel \
                -propType Propagation/TwoRayGround \
                -antType Antenna/OmniAntenna \
                -topoInstance $topo \
                -agentTrace ON \
                -routerTrace ON
create-god 3
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

$n0 label "tcp0"
$n1 label "sink1/tcp1"
$n2 label "sink2"

$n0 set X_ 50
$n0 set Y_ 50
$n0 set Z_ 0

$n1 set X_ 100
$n1 set Y_ 100
$n1 set Z_ 0

$n2 set X_ 600
$n2 set Y_ 600
$n2 set Z_ 0

$ns at 0.1 "$n0 setdest 50 50 15"
```

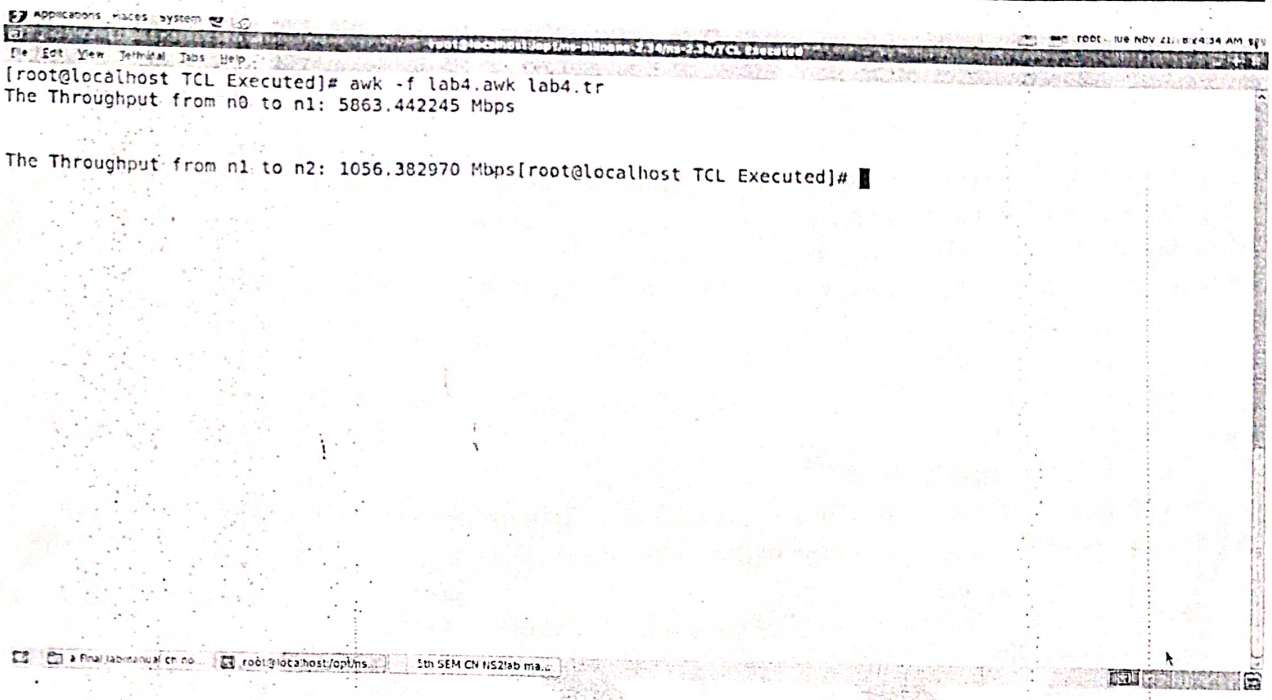
```
$ns at 0.1 "$n1 setdest 100 100 25"
$ns at 0.1 "$n2 setdest 600 600 25"

set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0

set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
set sink1 [new Agent/TCPSink]
$ns attach-agent $n1 $sink1
$ns connect $tcp0 $sink1
set tcp1 [new Agent/TCP]
$ns attach-agent $n1 $tcp1
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
set sink2 [new Agent/TCPSink]
$ns attach-agent $n2 $sink2
$ns connect $tcp1 $sink2
$ns at 5 "$ftp0 start"
$ns at 5 "$ftp1 start"

$ns at 100 "$n1 setdest 550 550 15"
$ns at 190 "$n1 setdest 70 70 15"
proc finish { } {
        global ns nf tf
        $ns flush-trace
        exec nam lab4.nam &
        close $tf
        exit 0
}
$ns at 250 "finish"
$ns run
```

AWK FILE(lab4.awk)

```
BEGIN{
    count1=0
    count2=0
    pack1=0
    pack2=0
    time1=0
    time2=0
}
```

```
{
    if($1=="r"&& $3=="_1_" && $4=="AGT")
    {
        count1++
        pack1=pack1+$8
        time1=$2
    }
    if($1=="r" && $3=="_2_" && $4=="AGT")
    {
        count2++
        pack2=pack2+$8
    }
}
END {
time2=$2;
printf("The Throughput from n0 to n1: %f Mbps", ((count1*pack1*8)/(time1*1000000)));
printf("The Throughput from n1 to n2: %f Mbps", ((count2*pack2*8)/(time2*1000000)));

}
```



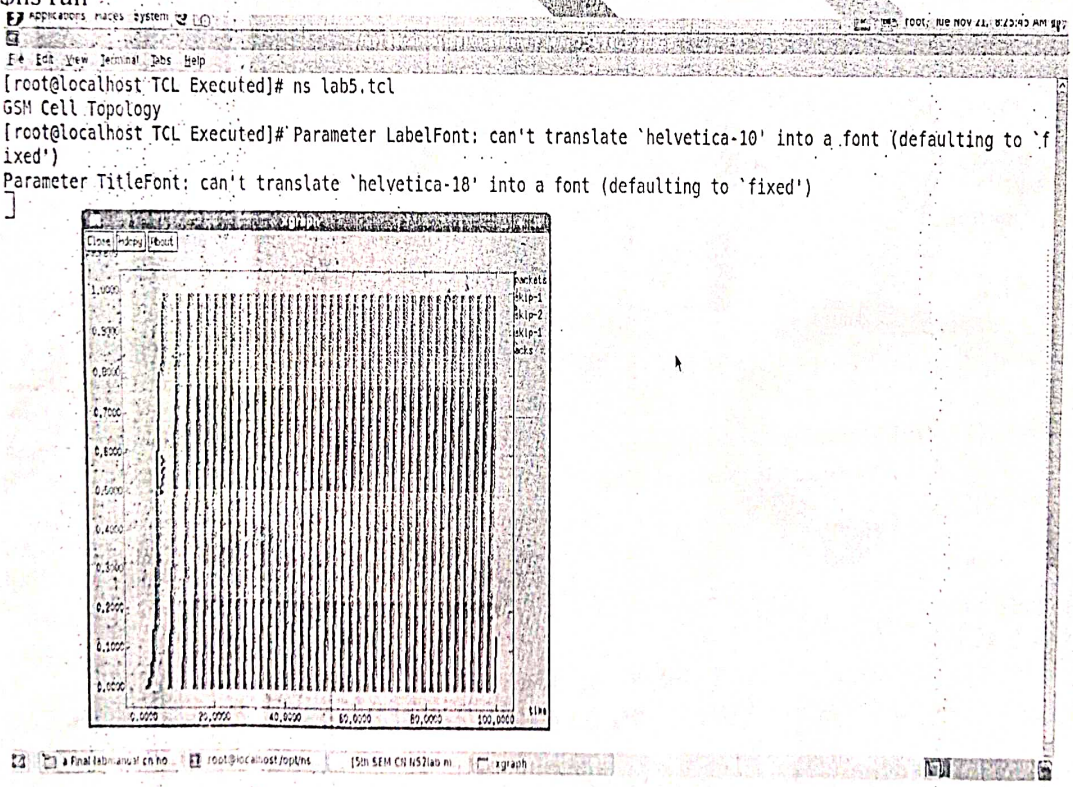


5.Implement and study the performance of GSM on NS2/NS3 (Using MAC layer) or equivalent environment.

```
set stop 100 ;#stop time
set type gsm  ;#type of link

set minth 30
set maxth 0
set adaptive 1

set flows 0
set window 30

set opt(wrap) 100
set opt(srcTrace) is
set opt(dstTrace) bs2

set bwDL(gsm) 9600
set propDL(gsm) .500

set ns [new Simulator]
set tf [open lab5.tr w]
$ns trace-all $tf

set nodes(is) [$ns node]
set nodes(ms) [$ns node]
set nodes(bs1) [$ns node]
```

```
set nodes(bs2) [$ns node]
set nodes(lp) [$ns node]

proc cell_topo {} {
global ns nodes
$ns duplex-link $nodes(lp) $nodes(bs1) 3Mbps 10ms DropTail
$ns duplex-link $nodes(bs1) $nodes(ms) 1 1 RED
$ns duplex-link $nodes(ms) $nodes(bs2) 1 1 RED
$ns duplex-link $nodes(bs2) $nodes(is) 3Mbps 10ms DropTail
puts "GSM Cell Topology"
}

proc set_link_params {t} {
        global ns nodes bwDL propDL
        $ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) duplex
        $ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) duplex

        $ns delay $nodes(bs1) $nodes(ms) $propDL($t) duplex
        $ns delay $nodes(bs2) $nodes(ms) $propDL($t) duplex

        $ns queue-limit $nodes(bs1) $nodes(ms) 10
        $ns queue-limit $nodes(bs2) $nodes(ms) 10
}

Queue/RED set adaptive_ $adaptive
Queue/RED set thresh_ $minth
Queue/RED set maxthresh_ $maxth
Agent/TCP set window_ $window

switch $type {
gsm -
cdma {cell_topo}
}
set_link_params $type
$ns insert-delayer $nodes(ms) $nodes(bs1) [new Delayer]
$ns insert-delayer $nodes(ms) $nodes(bs2) [new Delayer]

if {$flows ==0} {
        set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
        set ftp1 {[set tcp1] attach-app FTP]
        $ns at 0.8 "[set ftp1] start"
}


proc stop {} {
        global nodes opt tf
```

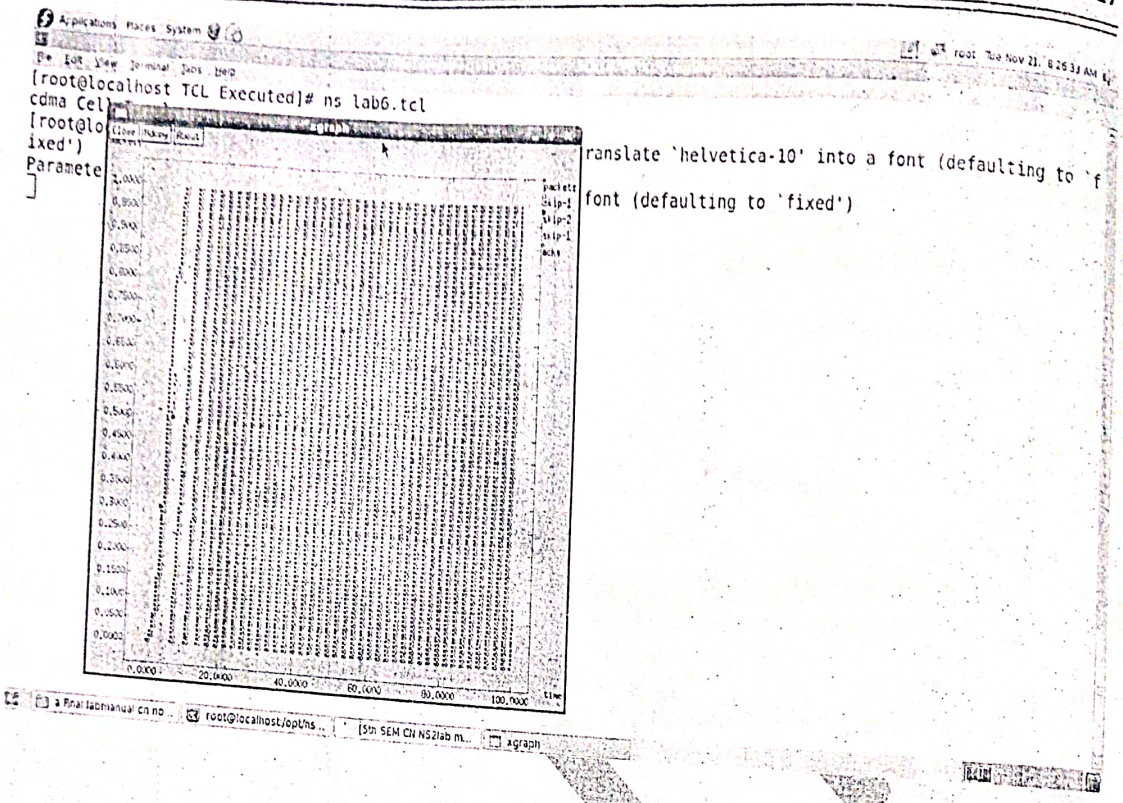
```
        set wrap $opt(wrap)
        set sid [$nodes($opt(srcTrace)) id]
        set did [$nodes($opt(dstTrace)) id]
        set a "lab5.tr"
        set GETRC "/opt/ns-allinone-2.34/ns-2.34/bin/getrc"
        set RAW2XG "/opt/ns-allinone-2.34/ns-2.34/bin/raw2xg"

exec $GETRC -s $sid -d $did -f 0 out.tr | \
   $RAW2XG -s 0.01 -m $wrap -r > plot.xgr

exec $GETRC -s $did -d $sid -f 0 out.tr | \
   $RAW2XG -a -s 0.01 -m $wrap  >> plot.xgr

exec xgraph -x time -y packets plot.xgr &
exit 0

}

$ns at $stop "stop"
$ns run
```



6.Implement and study the performance of CDMA on NS2/NS3 (Using stack called Call net) or equivalent environment.

```
set stop 100
set type cdma
set minth 30
set maxth 0
set adaptive 1
set flows 0
set window 30

set opt(wrap) 100
set opt(srcTrace) is
set opt(dstTrace) bs2

set bwDL(cdma) 384000
set propDL(cdma) .150

set ns [new Simulator]
set tf [open lab6.tr w]
$ns trace-all $tf

set nodes(is) [$ns node]
set nodes(ms) [$ns node]
set nodes(bs1) [$ns node]
set nodes(bs2) [$ns node]
set nodes(lp) [$ns node]

proc cell_topo {} {
global ns nodes
$ns duplex-link $nodes(lp)-$nodes(bs1) 3Mbps 10ms DropTail
$ns duplex-link $nodes(bs1) $nodes(ms) 1 1 RED
$ns duplex-link $nodes(ms) $nodes(bs2) 1 1 RED
$ns duplex-link $nodes(bs2) $nodes(is) 3Mbps 10ms DropTail
puts "cdma Cell Topolgy"
}

proc set_link_para {t} {
global ns nodes bwDL propDL
$ns bandwidth $nodes(bs1) $nodes(ms) $bwDL($t) duplex
$ns bandwidth $nodes(bs2) $nodes(ms) $bwDL($t) duplex

$ns delay $nodes(bs1) $nodes(ms) $propDL($t) duplex
$ns delay $nodes(bs2) $nodes(ms) $propDL($t) duplex

$ns queue-limit $nodes(bs1) $nodes(ms) 20
```

```
$ns queue-limit $nodes(bs2) $nodes(ms) 20
}

Queue/RED set adaptive_ $adaptive
Queue/RED set thresh_ $minth
Queue/RED set maxthresh_ $maxth

Agent/TCP set window_ $window

switch $type {
cdma {cell_topo}
}

set_link_para $type
$ns insert-delayer $nodes(ms) $nodes(bs1) [new Delayer]
$ns insert-delayer $nodes(ms) $nodes(bs2) [new Delayer]

if {$flows ==0} {
set tcp1 [$ns create-connection TCP/Sack1 $nodes(is) TCPSink/Sack1 $nodes(lp) 0]
set ftp1 [[set tcp1] attach-app FTP]
$ns at 0.8 "[set ftp1] start"
}

proc stop {} {
global nodes opt tf
set wrap $opt(wrap)
set sid [$nodes($opt(srcTrace)) id]
set did [$nodes($opt(dstTrace)) id]
set a "lab6.tr"
set GETRC "/opt/ns-allinone-2.34/ns-2.34/bin/getrc"
set RAW2XG "/opt/ns-allinone-2.34/ns-2.34/bin/raw2xg"

exec $GETRC -s $sid -d $did -f 0 out.tr  \
$RAW2XG -s 0.01 -m $wrap -r > plot.xgr

exec $GETRC -s $did -d $sid -f 0 out.tr | \
$RAW2XG -a -s 0.01 -m $wrap >> plot.xgr
exec xgraph -x time -y packets plot.xgr &
exit 0
}
$ns at $stop "stop"
$ns run
```


Applications Places System
File Edit View Terminal Tabs Help
root Tue Nov 21, 8:26:33 AM
[root@localhost TCL Executed]# ns lab6.tcl
cdma Cel
[root@lo
ixed')
Paramete
ranslate `helvetica-10' into a font (defaulting to `f
font (defaulting to `fixed')
0.0000
20.0000
40.0000
60.0000
80.0000
100.0000
time
a Final labmanual cn no
root@localhost/opt/ns ...
[5th SEM CN NS2lab m...
xgraph

data=1001
divisor=1011

PART- B (JAVA PROGRAMS)

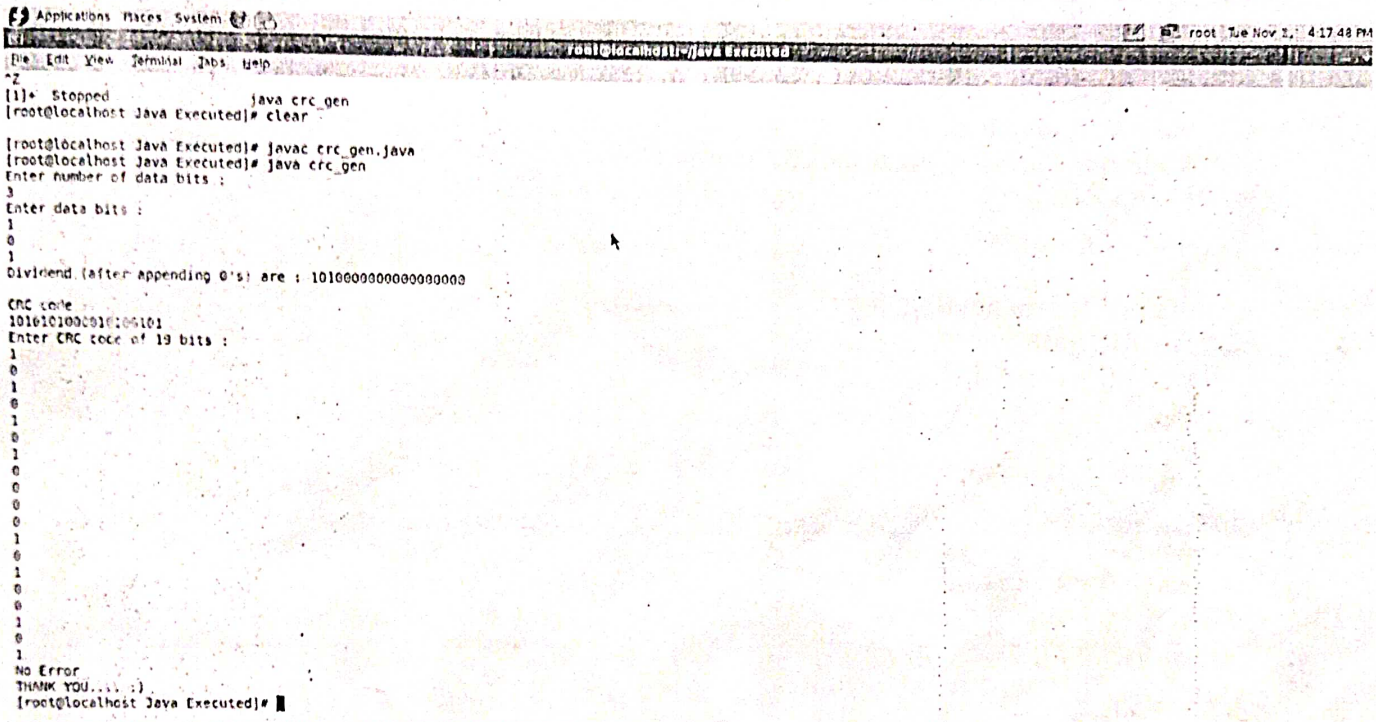
7. Write a program for error detecting code using CRC-CCITT (16- bits).

```
import java.io.*;
class crc_gen
{
 public static void main(String args[]) throws Exception, IOException
 {
 BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
 int[ ] data;
 int[ ] div;
 int[ ] divisor;
 int[ ] rem;
 int[ ] crc;
 int data_bits, divisor_bits, tot_length;

 System.out.println("Enter number of data bits : ");
 data_bits=Integer.parseInt(br.readLine());
 data=new int[data_bits];
 System.out.println("Enter data bits : ");
 for(int i=0; i<data_bits; i++)
 data[i]=Integer.parseInt(br.readLine());
 divisor_bits = 17;
 divisor = new int[]{1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1};
 tot_length=data_bits+divisor_bits-1;

 div=new int[tot_length];
 rem=new int[tot_length];
 crc=new int[tot_length];
 /*----------------- CRC GENERATION----------------------*/
 for(int i=0;i<data.length;i++)
 div[i]=data[i];

 System.out.print("Dividend (after appending 0's) are : ");
 for(int i=0; i< div.length; i++)
 System.out.print(div[i]);
 System.out.println();

 for(int j=0; j<div.length; j++){
rem[j] = div[j];
 }

 rem=divide(divisor, rem);
 for(int i=0;i<div.length;i++) //append dividend and ramainder
 {
```

$X^{16} + X^{12} + X^5 + X^0$

```
crc[i]=(div[i]^rem[i]);
}

System.out.println();
System.out.println("CRC code : ");
for(int i=0;i<crc.length;i++)
System.out.print(crc[i]);

/*----------ERROR DETECTION----------*/
System.out.println();
System.out.println("Enter CRC code of "+tot_length+" bits : ");
for(int i=0; i<crc.length; i++)
crc[i]=Integer.parseInt(br.readLine());

for(int j=0; j<crc.length; j++){
rem[j] = crc[j];
}
rem=divide(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");
}

System.out.println("THANK YOU.... :)");
}
static int[ ] divide(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;
```

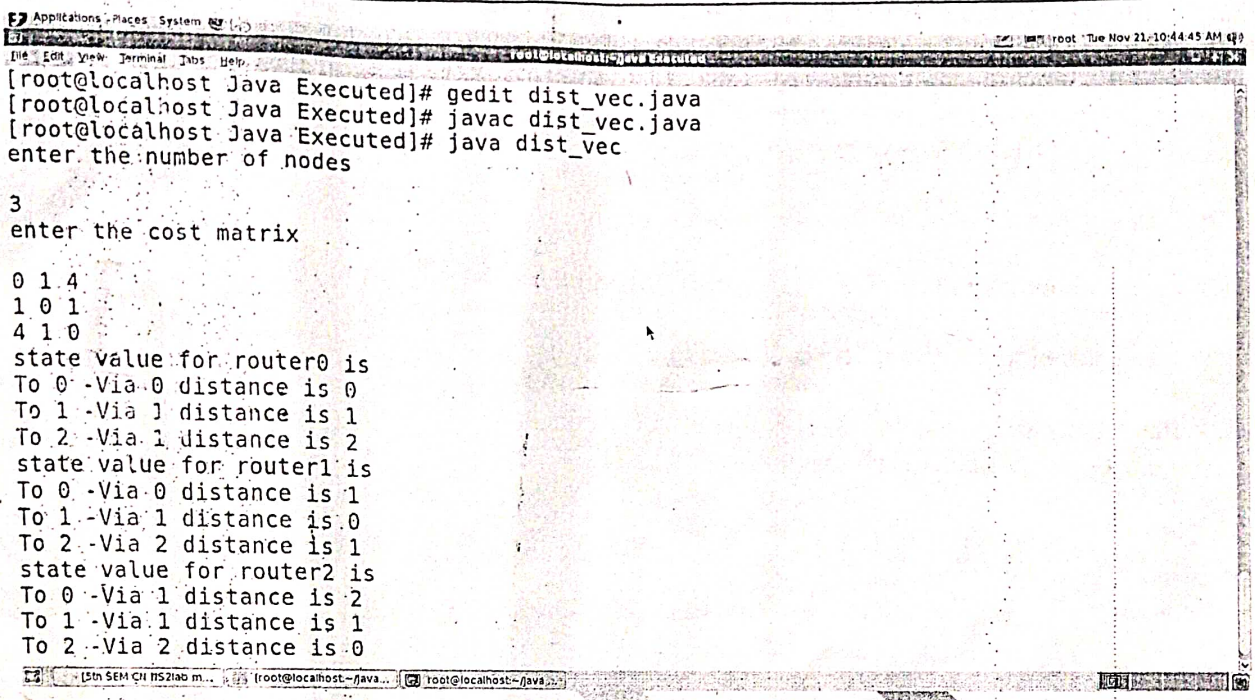
```
}
}
```



8.Write a program to find the shortest path between vertices using bellman ford algorithm.

```
import java.io.*;
import java.util.Scanner;
class dist_vec
{
public static void main(String args[])
{

        int dmat[ ][ ];
```

```
int dist[ ][ ];
int via[ ] [];
int n=0,i=0,j=0,k=0,count=0;
Scanner in = new Scanner(System.in);
System.out.println("enter the number of nodes\n");
n = in.nextInt();
dmat = new int[n][n];
dist = new int[n][n];
via = new int[n][n];
System.out.println("enter the cost matrix\n");
for(i=0;i<n;i++)
for(j=0;j<n;j++)
{
dmat[i][j] = in.nextInt();
dmat[i][i]=0;
dist[i][j]=dmat[i][j];
via[i][j]=j;
}
do
{
count=0;
for(i=0;i<n;i++)
for(j=0;j<n;j++)
for(k=0;k<n;k++)
if(dist[i][j]>dmat[i][k]+dist[k][j])
{
dist[i][j]=dist[i][k]+dist[k][j];
via[i][j]=k;
count++;
}
}while(count!=0);
for(i=0;i<n;i++)
{
System.out.println("state value for router"+i+" is");
for(j=0;j<n;j++)
{
System.out.println("To "+j+" -Via "+via[i][j]+" distance is "+dist[i][j]);
}
}
}
}
```



9.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. Implement the above program using as message queues or FIFOs as IPC channels.

TCP SERVER PROGRAM (TCPServer.java)

```
import java.net.*;
import java.io.*;
public class TCPServer
{
  public static void main(String args[]) throws Exception
  {
    // establishing the connection with the server
    ServerSocket sersock = new ServerSocket(4000);
    System.out.println("Server ready for connection");
    Socket sock = sersock.accept();          // binding with port: 4000
    System.out.println("Connection is successful and wating for chatting");
    // reading the file name from client
    InputStream istream = sock.getInputStream( );
```

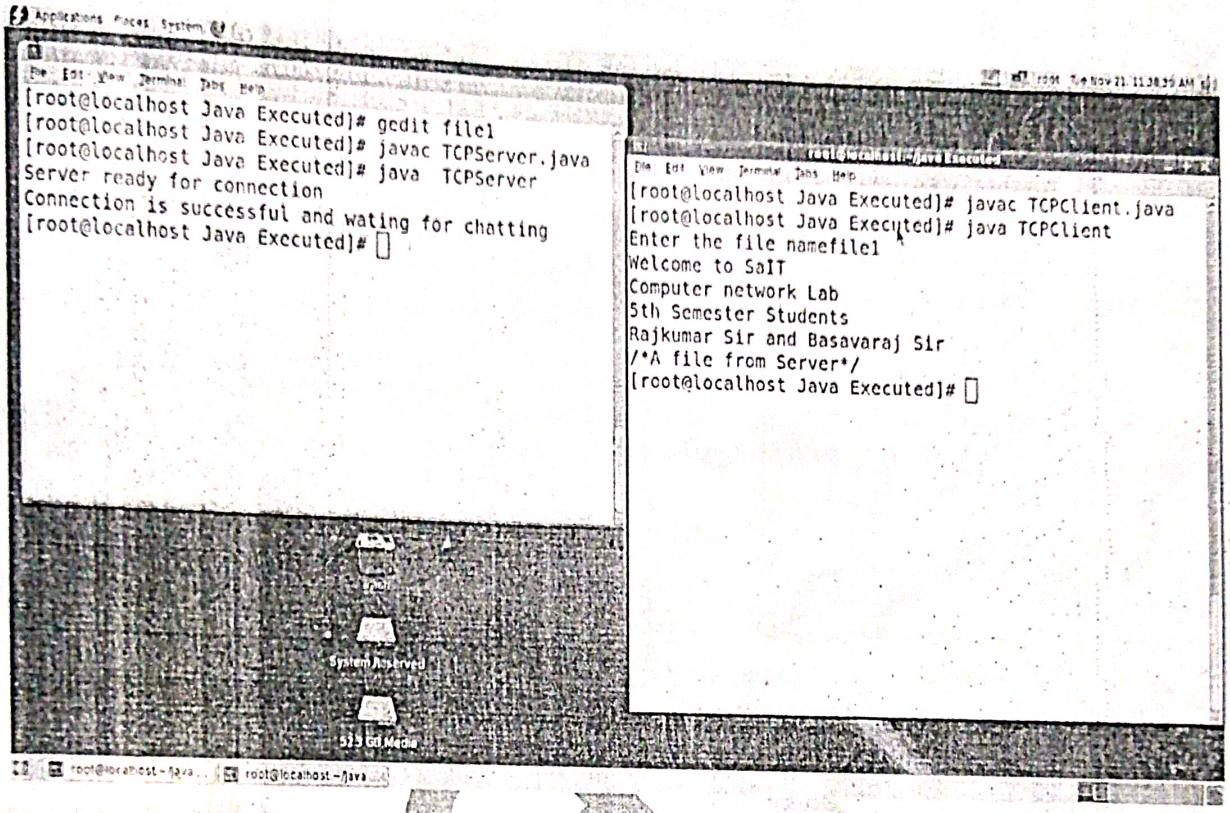
```
    BufferedReader fileRead =new BufferedReader(new InputStreamReader(istream));
    String fname = fileRead.readLine( );
    // reading file contents
    BufferedReader contentRead = new BufferedReader(new FileReader(fname) );
    // keeping output stream ready to send the contents
    OutputStream ostream = sock.getOutputStream( );
    PrintWriter pwrite = new PrintWriter(ostream, true);
    String str;
    while((str = contentRead.readLine()) != null) // reading line-by-line from file
    {
       pwrite.println(str);   // sending each line to client
    }
    sock.close(); sersock.close();     // closing network sockets
    pwrite.close(); fileRead.close(); contentRead.close();
  }
}
```

TCP CLIENT PROGRAM (TCPClient.java)

```
import java.net.*;
import java.io.*;
public class TCPClient
{
 public static void main( String args[ ] ) throws Exception
 {
  Socket sock = new Socket( "127.0.0.1", 4000);
    //reading the file name from keyboard. Uses input stream
    System.out.print("Enter the file name");
    BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));
    String fname = keyRead.readLine();
   // sending the file name to server. Uses PrintWriter
    OutputStream ostream = sock.getOutputStream( );
    PrintWriter pwrite = new PrintWriter(ostream, true);
    pwrite.println(fname);
    // receiving the contents from server. Uses input stream
    InputStream istream = sock.getInputStream();
    BufferedReader socketRead = new BufferedReader(new InputStreamReader(istream));
    String str;
    while((str = socketRead.readLine()) != null) // reading line-by-line
    {
      System.out.println(str);
    }
    pwrite.close(); socketRead.close(); keyRead.close();
```

* Implements all the print methods found in printstream.

```
}
}
```



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

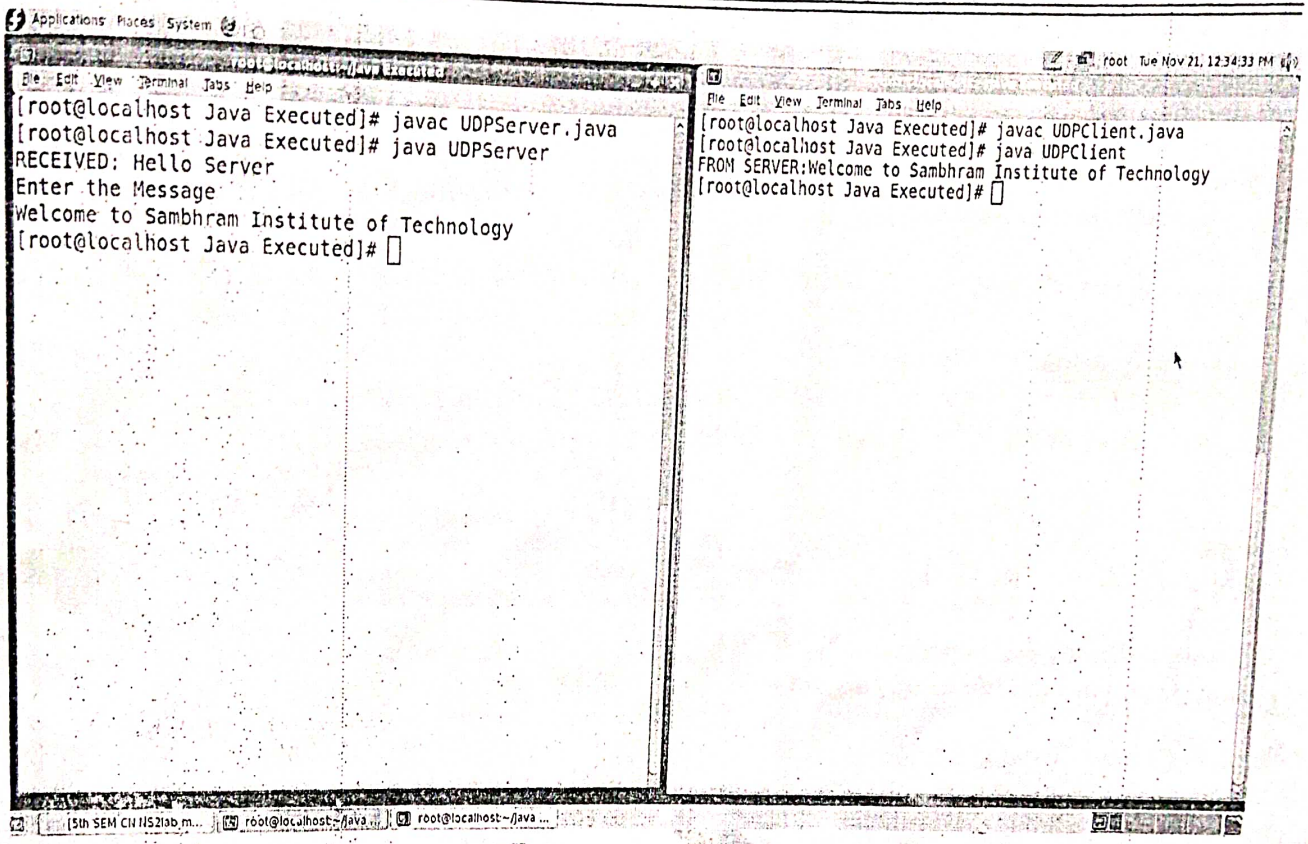
UDP SERVER PROGRAM (UDPServer.java)

```
import java.io.*;
import java.net.*;
class UDPServer
{
 public static void main(String args[]) throws Exception
   {
    DatagramSocket serverSocket = new DatagramSocket(9876);
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    byte[] receiveData = new byte[1024];
    byte[] sendData = new byte[1024];
    DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
    serverSocket.receive(receivePacket);
```

```
    String sentence = new String( receivePacket.getData());
    System.out.println("RECEIVED: " + sentence);
     InetAddress IPAddress = receivePacket.getAddress();
     int port = receivePacket.getPort();
    System.out.println("Enter the Message");
    String data = br.readLine();
    sendData=data.getBytes();
    DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
IPAddress, port);
    serverSocket.send(sendPacket);
    serverSocket.close();
  }
}
```

UDP CLIENT PROGRAM (UDPClient.java)

```
import java.net.*;
class UDPClient
{
 public static void main(String args[]) throws Exception
import java.io.*;
 {
   BufferedReader in = 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];
   String sentence = "Hello Server";
   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());
   System.out.println("FROM SERVER:" + modifiedSentence);
   clientSocket.close();
  }
}
```

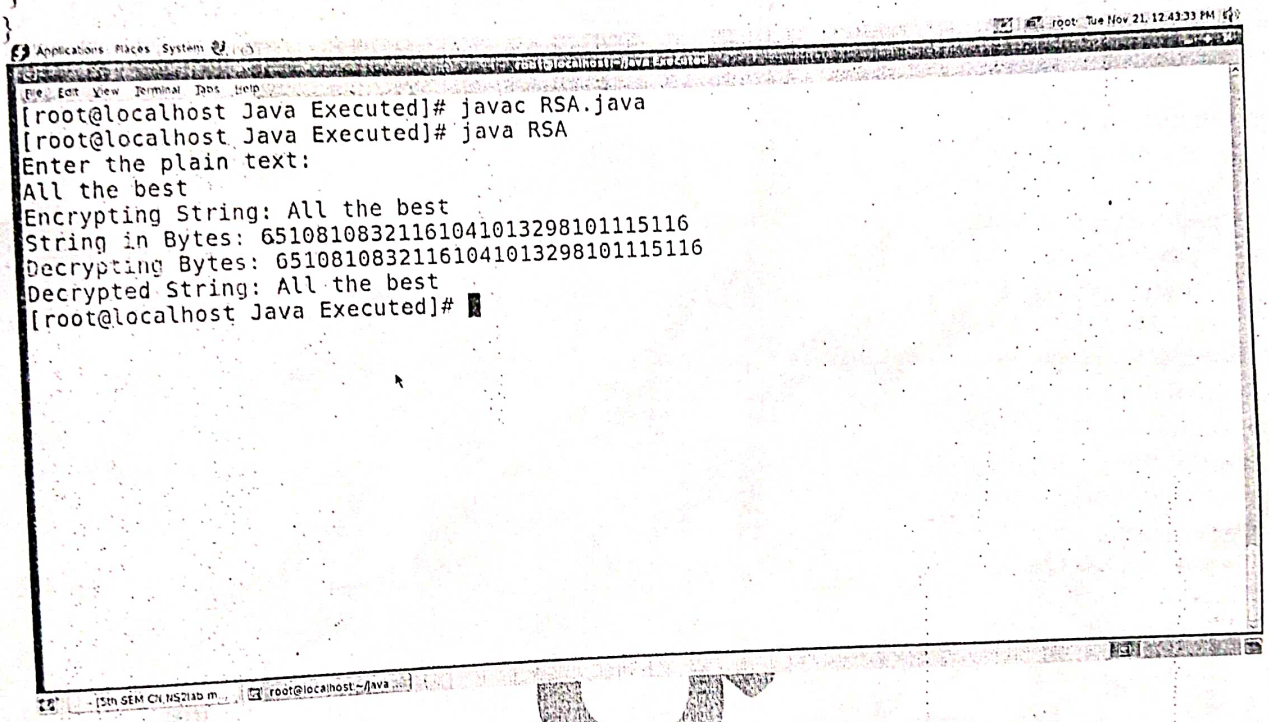


11.Write a program for simple RSA algorithm to encrypt and decrypt the data.

```
this.e = e; import java.io.DataInputStream;
import java.io.IOException;
import java.math.BigInteger;
import java.util.Random;
public class RSA
{
 private BigInteger p;
 private BigInteger q;
 private BigInteger N;
private BigInteger phi;
 private BigInteger e;
 private BigInteger d;
 private int bitlength = 1024;
 private Random r;
 public RSA()
 {
 r = new Random();
```

```
p = BigInteger.probablePrime(bitlength, r);
q = BigInteger.probablePrime(bitlength, r);
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);
}
d = e.modInverse(phi);
}
public RSA(BigInteger e, BigInteger d, BigInteger N)
{

this.d = d;
this.N = N;
}
@SuppressWarnings("deprecation")
public static void main(String[] args) throws IOException
{
RSA rsa = new RSA();
DataInputStream in = new DataInputStream(System.in);
String teststring;
System.out.println("Enter the plain text:");
teststring = in.readLine();
System.out.println("Encrypting String: " + teststring);
System.out.println("String in Bytes: " + bytesToString(teststring.getBytes()));
// encrypt
byte[] encrypted = rsa.encrypt(teststring.getBytes());
// decrypt
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)
{
String test = "";
for (byte b : encrypted)
{
test += Byte.toString(b);
}
return test;
}
// Encrypt message
public byte[] encrypt(byte[] message)
```

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



12.Write a program for congestion control using leaky bucket algorithm.

```
import java.lang.*;
import java.util.Random;
import java.io.*;
import java.util.Scanner;
class leaky_bucket
{
public static void main(String args[])
{
int drop=0,mini,nsec,p_remain=0;
int o_rate,b_size,i,packet[];
packet = new int[100];
Scanner in = new Scanner(System.in);
System.out.println("Enter bucket size:");
```

```
b_size = in.nextInt();
System.out.println("Enter the output rate:");
o_rate = in.nextInt();
System.out.println("Enter the number of seconds you want to simulate:");
nsec = in.nextInt();
Random rand = new Random();
for(i=0;i<nsec;i++)
packet[i]=((rand.nextInt(9)+1)*10);
System.out.println("Seconds|packets received|packets sent|packets left|packets dropped");
System.out.println("-----------------------------------------------------------");
for(i=0;i<nsec;i++)
{
p_remain+=packet[i];
if(p_remain>b_size)
{
drop=p_remain-b_size;
p_remain=b_size;
System.out.print(i+1+" ");
System.out.print(packet[i]+" ");
mini=Math.min(p_remain,o_rate);
System.out.print(mini+" ");
p_remain=p_remain-mini;
System.out.print(p_remain+" ");
System.out.print(drop+" ");
System.out.println();
drop=0;
}
}
while(p_remain!=0)
{
if(p_remain>b_size)
{
drop=p_remain-b_size;
p_remain=b_size;
}
mini=Math.min(p_remain,o_rate);
System.out.print(" "+p_remain+" "+mini);
p_remain=p_remain-mini;
System.out.println(p_remain+" "+drop);
drop=0;
}
}
}
```

```
[root@localhost Java Executed]# javac leaky_bucket.java
[root@localhost Java Executed]# java leaky_bucket
Enter bucket size:
5
Enter the output rate:
3
Enter the number of seconds you want to simulate:
3
Seconds|packets received|packets sent|packets left|packets dropped
-------------------------------------------------------------------
1 80 3 2 75
2 90 3 2 87
3 30 3 2 27
 2 20 0
[root@localhost Java Executed]# gedit leaky_bucket.java
```

**********ALL THE BEST 5TH SEM ISE STUDENTS**********