PROGRAM 1

Prog1.tcl

```
set ns [new Simulator]
$ns color 0 Blue
set nf [open Prog1.nam w]
$ns namtrace-all $nf
set nd [open Prog1.tr w]
$ns trace-all $nd
proc finish {} {
global ns nf nd
$ns flush-trace
close $nf
close $nd
exec nam Prog1.nam &
exit 0
}
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
$ns duplex-link $n0 $n1 5Mb 5ms DropTail
$ns duplex-link $n1 $n2 1Mb 5ms DropTail
$ns queue-limit $n1 $n2 8
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 1024
$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
```

Prog1.awk

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

Program 2

Prog2.tcl

```
set ns [new Simulator]
set nf [open Prog2.nam w]
$ns namtrace-all $nf
set tf [open Prog2.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]

$ns duplex-link $n0 $n4 15Mb 1ms DropTail
$ns duplex-link $n1 $n4 5Mb 1ms DropTail
$ns duplex-link $n2 $n4 20Mb 1ms DropTail
$ns duplex-link $n3 $n4 200Kb 1ms DropTail
$ns duplex-link $n4 $n5 1Mb 1ms DropTail

set p1 [new Agent/Ping]
$ns attach-agent $n0 $p1
$p1 set packetSize_ 500
$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_ 300
$p3 set interval_ 0.00001
set p4 [new Agent/Ping]
$ns attach-agent $n3 $p4
set p5 [new Agent/Ping]
$ns attach-agent $n5 $p5
$ns queue-limit $n0 $n4 3
$ns queue-limit $n2 $n4 2
$ns queue-limit $n4 $n5 1

Agent/Ping instproc recv {from rtt} {
$self instvar node_
puts "node [$node_ id] recieved 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 Prog2.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
```

Prog2.awk

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

Program 3

Prog3.tcl

```
set ns [new Simulator]
set tf [open prog3.tr w]
$ns trace-all $tf
set nf [open prog3.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]
$ftp2 attach-agent $tcp2
$ftp2 set packetSize_ 600
$ftp2 set interval_ 0.001
set sink3 [new Agent/TCPSink]
$ns attach-agent $n3 $sink3
$ns 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 prog3.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
```

Prog3.awk

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

Program 4

Prog4.tcl

```
set ns [new Simulator]
set tf [open prog4.tr w]
$ns trace-all $tf
set topo [new Topography]
$topo load_flatgrid 1000 1000
set nf [open prog4.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 550 500 15"
$ns at 190 "$n1 setdest 70 70 15"
proc finish { } {
global ns nf tf
$ns flush-trace
exec nam prog4.nam &
close $tf
exit 0
}
$ns at 250 "finish"
$ns run
```

Prog4.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
                time2=$2
```

```
        }
}
END { printf("The Throghput from n0 to n1 : %f Mbps \n",((count1*pack1*8)/(time1*1000000)));
printf("The Throghput from n1 to n2 : %f Mbps \n",((count2*pack2*8)/(time2*1000000)));
}
```

Program 5

5M.tcl

```
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 out.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 "out.tr"
        set GETRC "../../../bin/getrc"
        set RAW2XG "../../../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
```

Program 6

6M.tcl

```
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 out.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 "out.tr"
        set GETRC "../../../bin/getrc"
        set RAW2XG "../../../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
```

Program 7

Crc_gen.java

```
import java.io.*;
class crc_gen
{
        public static void main(String args[]) throws IOException
        {
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        int data[];
        int div[], divisor[], rem[], crc[], 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());
        System.out.println("Enter number of bits in divisor: ");
        divisor_bits=Integer.parseInt(br.readLine());
        divisor=new int[divisor_bits];
        System.out.println("Enter divisor bits: ");
        for(int i=0; i<divisor_bits; i++)
                divisor[i]=Integer.parseInt(br.readLine());
        System.out.print("Data bits are: ");
        for(int i=0;i<data_bits; i++)
                System.out.print(data[i]);
        System.out.println();
        System.out.print("Divisor bits are: ");
        for(int i=0; i<divisor_bits;i++)
                System.out.print(divisor[i]);
        System.out.println();
        tot_length = data_bits + divisor_bits - 1;
        div=new int[tot_length];
        rem=new int[tot_length];
        crc=new int[tot_length];
        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(div,divisor,rem);
        for(int i=0; i<div.length;i++)
                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]);
        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(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");
        }
        System.out.println("THANK YOU.....");
        }
        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;
        }
}
```

Program 8

BellmanFord.java

```
import java.util.Scanner;
public class BellmanFord
{
	private int D[];
	private int num_ver;
	public static final int MAX_VALUE=999;
	public BellmanFord(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 edge 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();
		BellmanFord b=new BellmanFord(num_ver);
		b.BellmanFordEvaluation(source,A);
		scanner.close();
	}
}
```

Program 9

ContentsClient.java

```
import java.net.*;
import java.io.*;
public class ContentsClient
{
  public static void main( String args[ ] ) throws Exception
  {
    Socket sock = new Socket( "127.0.0.1", 4000);
    System.out.print("Enter the file name");
    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();
  }
}
```

ContentsServer.java

```
import java.net.*;
import java.io.*;
public class ContentsServer
{
  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 wating 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();
  }
}
```

Program 10

MyClient.java

```
import java.net.*;
import java.io.*;
class MyClient
{
          public static void main(String args[]) throws Exception
          {
```

```
                DatagramSocket ds = new DatagramSocket(16000);
                byte buffer[] = new byte[100];
                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
                System.out.println("Chat Application Started ! .... Type message to send and bye to quit");
                String str = "",str2;
                do
                {
                        System.out.print("Client says :");
                        str2 = br.readLine();
                        java.util.Arrays.fill(buffer, (byte)0);
                        for(int i=0; i<str2.length(); i++)
                                buffer[i] = (byte)str2.charAt(i);
                ds.send(new DatagramPacket(buffer, buffer.length, InetAddress.getLocalHost(), 15000));
                        if(!str2.equals("bye"))
                        {
                                System.out.print("Server says: ");
                                DatagramPacket p = new DatagramPacket(buffer, buffer.length);
                                ds.receive(p);
                                str = "";
                                str = new String(p.getData());
                                System.out.println(str);
                        }
                }while(!str2.equals("bye"));
                System.out.println("Closing chat Application");
                ds.close();
        }
}
```

MyServer.java

```
import java.net.*;
import java.io.*;
class MyServer
{
        public static void main(String args[]) throws Exception
        {
                DatagramSocket ds = new DatagramSocket(15000);
                byte buffer[] = new byte[100];
                BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
                String str2;
                do
                {
                        DatagramPacket p = new DatagramPacket(buffer, buffer.length);
                        ds.receive(p);
                        String str = new String(p.getData());
                        System.out.println("Client says:"+str);
                        System.out.print("Server says :");
                        str2 = br.readLine();
                        java.util.Arrays.fill(buffer, (byte)0);
                        for(int i=0; i<str2.length(); i++)
```

```
                    buffer[i] = (byte)str2.charAt(i);
            ds.send(new DatagramPacket(buffer, buffer.length, InetAddress.getLocalHost(), 16000));
            }while(!str2.equals("bye"));
            System.out.println("Closing chat Application");
            ds.close();
        }
}
```

Program 11

RSAkeygen.java

```
import java.util.*;
import java.math.BigInteger;
import java.lang.*;
class RSAkeygen
{
        public static void main(String args[])
        {
                Random rand1=new Random(System.currentTimeMillis());
                Random rand2=new Random(System.currentTimeMillis()*10);
                int pubkey=Integer.parseInt(args[0]);
                BigInteger bigB_p=BigInteger.probablePrime(32,rand1);
                BigInteger bigB_q=BigInteger.probablePrime(32,rand2);
                BigInteger bigB_n=bigB_p.multiply(bigB_q);
                BigInteger bigB_p_1=bigB_p.subtract(new BigInteger("1"));
                BigInteger bigB_q_1=bigB_q.subtract(new BigInteger("1"));
                BigInteger bigB_p_1_q_1=bigB_p_1.multiply(bigB_q_1);
                while(true)
                {
                        BigInteger BigB_GCD=bigB_p_1_q_1.gcd(new BigInteger(""+pubkey));
                        if(BigB_GCD.equals(BigInteger.ONE))
                                break;
                        pubkey++;
                }
                BigInteger bigB_pubkey=new BigInteger(""+pubkey);
                BigInteger bigB_prvkey=bigB_pubkey.modInverse(bigB_p_1_q_1);
                System.out.println("public key:"+bigB_pubkey+","+bigB_n);
                System.out.println("private key:"+bigB_prvkey+","+bigB_n);
        }
}
```

RSAEncDec.java

```
import java.math.BigInteger;
import java.util.*;
class RSAEncDec
{
        public static void main(String args[])
        {
```

```
                BigInteger bigB_pubkey=new BigInteger(args[0]);
                BigInteger bigB_prvkey=new BigInteger(args[1]);
                BigInteger bigB_n=new BigInteger(args[2]);
                int asciiVal=Integer.parseInt(args[3]);
                BigInteger bigB_val=new BigInteger(""+asciiVal);
                BigInteger bigB_cipherVal=bigB_val.modPow(bigB_pubkey,bigB_n);
                System.out.println("Cipher text: "+bigB_cipherVal);
                BigInteger bigB_plainVal=bigB_cipherVal.modPow(bigB_prvkey,bigB_n);
                int plainVal=bigB_plainVal.intValue();
                System.out.println("Plain text:"+plainVal);
        }
}
```

Program 12

Leaky.java

```
import java.io.*;
import java.util.*;
class Queue
{
        int q[],f=0,r=0,size;
        void insert(int n)
        {
                Scanner sc=new Scanner(System.in);
                q=new int[10];
                for(int i=0;i<n;i++)
                {
                        System.out.print("\nEnter "+i+" element: ");
                        int ele=sc.nextInt();
                        if(r+1 > 10)
                        {
                                System.out.println("\nQueue is full \nLost Packet: "+ele);
                                break;
                        }
                        else
                        {
                                r++;
                                q[i]=ele;
                        }
                }
        }
        void delete()
        {
                Thread t=new Thread();
                if(r==0)
                        System.out.print("\nQueue empty");
                else
                {
```

```
                        for(int i=f;i<r;i++)
                        {
                                try
                                {
                                        t.sleep(1000);
                                }
                                catch(Exception e) {}
                                System.out.print("\nLeaked Packet: "+q[i]);
                                f++;
                        }
                }
                System.out.println();
        }
}
class Leaky extends Thread
{
        public static void main(String ar[])  throws Exception
        {
                Queue q=new Queue();
                Scanner sc=new Scanner(System.in);
                System.out.println("\nEnter the packets to be sent :");
                int size=sc.nextInt();
                q.insert(size);
                q.delete();
        }
}
```