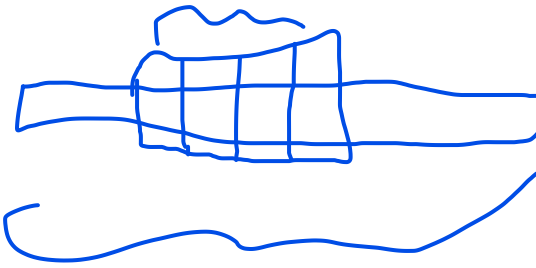
```
//Implement three nodes point – to – point network with duplex links between them. Set the
queue size, vary the bandwidth, and find the number of packets dropped.


BEGIN { c=0;
}
{   if($1=="d")
        {
                c++;
                printf("%s\t%s\n",$5,$11);
        }
}
END
{
printf("The number of packets dropped =%d\n",c);
}
//java -jar NSG2.1.jar
//gedit exam1.awk
//ns exam1.tcl
//awk -f exam1.awk exam1.tr
```

```
//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.


BEGIN { c=0;
}
{
        if($1=="d")
        {
                c++;
        }
}

END {
printf("Total n of %s pkts dropped due to congestion =%d\n",$5,c);
}


//java -jar NSG2.1.jar
//gedit exam1.awk
//ns exam1.tcl
//awk -f exam1.awk exam1.tr
```

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

BEGIN{
}
{
        if($6=="cwnd_")
                {printf("%f\t%f\t\n",$1,$7);
                }
}
END{
}
//java -jar NSG2.1.jar
//gedit exam1.awk
//ns exam1.tcl
//awk -f exam1.awk exam1.tr

//awk -f exam1.awk file1.tr >a1

//awk -f exam1.awk file2.tr >a2
//xgraph a1 a2
```

```
/*
Develop a program to implement a sliding window protocol in the data link layer.
*/

import java.util.Scanner;
public class exp5
{
public static void main(String[] args)
{
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter total frames to send: ");
        int tf = sc.nextInt();
        System.out.print("Enter the window size: ");
        int ws = sc.nextInt();
        int sent=0,finw;String ack;

        while (sent < tf)
        {
            finw = Math.min(ws,tf-sent);
            System.out.println("\nSending Frames:");
            for (int i = 0; i < finw; i++)
                System.out.println("Sent Frame " + (sent + i));

            int i = 0;
            while (i<finw)
            {
                System.out.print("Is ACK received? for Frame :" + (sent + i) + "  (y/n):
");
                ack = sc.next();
                if (ack.equals("y"))
                    System.out.println("ACK received for Frame "+( sent+ i++ ));
                else
                {   System.out.println("Resending.." + (sent + i));
                    break;
                }
            }
            sent =sent+ i;
        }
        System.out.println("\nAll frames sent successfully.");
        sc.close();
}
}
```



```java
/*
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.
*/

import java.io.*;
import java.net.*;
public class exp7_Server
{
    public static void main(String[] args)
    {

        String line,fn;
        try (ServerSocket ss = new ServerSocket(5000))
        {
            System.out.println("Server started...");
            Socket s = ss.accept();
            System.out.println("connected.");

            BufferedReader in = new BufferedReader(new
InputStreamReader(s.getInputStream()));
            PrintWriter out = new PrintWriter(s.getOutputStream(), true);
            fn = in.readLine();

            System.out.println("Requested file: " + fn);
            File f = new File(fn);
            if (f.exists() && f.isFile())
            {
                BufferedReader fr = new BufferedReader(new FileReader(f));
                while ((line = fr.readLine()) != null)
                    out.println(line);

                fr.close();
                out.println("EOF");
            }
             else
                out.println("File not found!");

            s.close();
            System.out.println("Connection closed.");
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```java
import java.io.*;
import java.net.*;
public class exp7_Client {
    public static void main(String[] args)
    {
        String fn;String line;

        try (Socket s = new Socket("localhost", 5000))
        {

         BufferedReader in = new BufferedReader(new
InputStreamReader(s.getInputStream()));
         PrintWriter out = new PrintWriter(s.getOutputStream(), true);
         BufferedReader cmd = new BufferedReader(new InputStreamReader(System.in));
            System.out.print("Enter file name: ");
            fn = cmd.readLine();
            out.println(fn);

            System.out.println("\nFile contents :\n");

            while ((line = in.readLine()) != null)
            {
                if (line.equals("EOF")) break;
                System.out.println(line);
            }
            s.close();
        }
        catch (Exception e){
            e.printStackTrace();}

    }
}
```

```java
/*
Develop a program on a datagram socket for client/server to display the messages on
client
side, typed at the server side.
*/
import java.net.*;
import java.util.*;
public class exp8_UDPServer {
    public static void main(String[] args) throws Exception
    {
        Scanner sc = new Scanner(System.in);

        DatagramSocket ds = new DatagramSocket();
        InetAddress a = InetAddress.getByName("localhost");

        System.out.println("Type messages");
        while (true)
        {
            String msg = sc.nextLine();
            byte[] bf = msg.getBytes();
            DatagramPacket pk = new DatagramPacket(bf,bf.length,a,5000);
            ds.send(pk);
            if (msg.equalsIgnoreCase("exit"))
                break;
        }
        s.close();
        sc.close();
    }
}
```

```java
import java.net.*;

public
class exp8_UDPClient {
 public
  static void main(String[] args) throws Exception
  {
    DatagramSocket ss = new DatagramSocket(5000);
    byte[] bf = new byte[1024];
    System.out.println("waiting for msg...");
    while (true) {
      DatagramPacket pk = new DatagramPacket(bf, bf.length);
      ss.receive(pk);
      String msg = new String(pk.getData(), 0, pk.getLength());
      System.out.println("Server says: " + msg);
      if (message.equalsIgnoreCase("exit")) {
        System.out.println("Client stopped.");
        break;
      }
    }
    s.close();
  }
}
```

```java
/*
Develop a program for congestion control using a leaky bucket algorithm.
*/
import java.util.Scanner;
public class exp10 {
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        int bk,or,n,cs=0,i,sent;


        System.out.print("Enter bucketCapacity,outputRate,n of packets ");
        bk = sc.nextInt();
        or = sc.nextInt();
        n = sc.nextInt();
        int[] packets = new int[n];
        System.out.println("Enter packet sizes :");
        for (i = 0; i < n; i++)
            packets[i] = sc.nextInt();

        for (i = 0; i < n; i++)
        {
            System.out.println("\nPacket "+(i+1)+" of size "+packets[i]+" arrived.");
            if (packets[i] + cs>bk)
                System.out.println("Bucket overflow");
             else
             {
                cs += packets[i];
                System.out.println("Packet added to bucket. Current bucket size: "+cs);
             }
            if (cs > 0)
            {
                sent = Math.min(or, cs);
                cs =cs - sent;
                System.out.println("Transmitted "+sent+" packets. Remaining in bucket: " +
cs);
            }
        }
        while (cs > 0)
        {
            sent = Math.min(or, cs);
            cs -= sent;
            System.out.println("Transmitted "+sent+" packets. Remaining in bucket: "+cs);
        }
        System.out.println("\nSimulation complete.");
        sc.close();
    }
}
```