```
#include <stdio.h>
int main()
{
int i=0,j=0,count=0;
char frame[100],stuffframe[200]="\0";
printf("enter the bits : ");
scanf("%s",frame);
printf("\nAfter bit stuffing \n");
strcpy(stuffframe,"01111110");
j=strlen(stuffframe);
for(i=0;frame[i]; i++)
{
                if(frame[i]=='1')
                count++;
                else
                count=0;
                stuffframe[j++]=frame[i];
                if(count==5)
                {
                stuffframe[j++]='0';
                count=0;
                }
}
strcat(stuffframe,"01111110");
printf("%s",stuffframe);
return  0; }
```

```
char stuff
#include <stdio.h>
#include <string.h>
int main( )
{ int i=0,j=0;
char frame[20], stuffframe[50]="\0";
printf("Enter Frame\n");
scanf("%s", frame);
/* Insert the DLESTX Flag*/
strcpy(stuffframe,"DLESTX");
j=strlen("DLESTX");
for(i=0; i<strlen(frame); i++) {
if(frame [i]=='D' && frame [i+1]=='L' && frame [i+2]=='E') {
stuffframe [j++]='D';
stuffframe [j++]='L';
stuffframe [j++]='E';
stuffframe [j++]= frame[i++];
stuffframe [j++]= frame [i++];
stuffframe [j++]= frame [i];
}
else
stuffframe [j++]= frame [i]; }
strcat(stuffframe,"DLEETX");
printf("\nFrame after stuffing:\n");
printf("%s", stuffframe);
return 0; }
```

```
stopwait
#include<stdio.h>
#include<stdlib.h>
int main()
{
int i,j,noframes,x,x2;
  i=1;
  j=1;
printf("Enter No of frames\t:");
scanf("%d", &noframes);
while(noframes>0)
{ printf("\nsending frame\t:%d",i);
   x=rand()%15;
if(x%5==0) {
 for(x2=1;x2<2;x2++) {
printf("\nwaiting for %d seconds for acknolegement",x2);
sleep(x2);
}
printf("\n sending frames %d\n ",i);
                x=rand()%10; }
printf("\nAcknoledgement received for frame %d\n",j);
noframes-=1;
i++;
j++;
} printf("END of stop and wait protocol\n"); }
```

```
link state routing
set ns [new Simulator]
set nr [open thro.tr w]
$ns trace-all $nr
set nf [open thro.nam w]
$ns namtrace-all $nf
      proc finish { } {
      global ns nr nf
      $ns flush-trace
      close $nf
      close $nr
      exec nam thro.nam &
                exit 0      }
for { set i 0 } { $i < 12} { incr i 1 } {
set n($i) [$ns node]}

for {set i 0} {$i < 8} {incr i} {
$ns duplex-link $n($i) $n([expr $i+1]) 1Mb 10ms
DropTail  }

$ns duplex-link $n(0) $n(8) 1Mb 10ms DropTail
$ns duplex-link $n(1) $n(10) 1Mb 10ms DropTail
$ns duplex-link $n(0) $n(9) 1Mb 10ms DropTail
$ns duplex-link $n(9) $n(11) 1Mb 10ms DropTail
$ns duplex-link $n(10) $n(11) 1Mb 10ms DropTail
$ns duplex-link $n(11) $n(5) 1Mb 10ms DropTail
set udp0 [new Agent/UDP]
$ns attach-agent $n(0) $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 set packetSize_ 500
$cbr0 set  interval_ 0.005
$cbr0 attach-agent $udp0
set null0 [new Agent/Null]
$ns attach-agent $n(5) $null0
$ns connect $udp0 $null0
set udp1 [new Agent/UDP]
$ns attach-agent $n(1) $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 set packetSize_ 500
$cbr1 set  interval_ 0.005
$cbr1 attach-agent $udp1
set null0 [new Agent/Null]
$ns attach-agent $n(5) $null0
$ns connect $udp1 $null0
$ns rtproto LS
$ns rtmodel-at 10.0 down $n(11) $n(5)
$ns rtmodel-at 15.0 down $n(7) $n(6)
$ns rtmodel-at 30.0 up $n(11) $n(5)
$ns rtmodel-at 20.0 up $n(7) $n(6)
$udp0 set fid_ 1
$udp1 set fid_ 2
$ns color 1 Red
$ns color 2 Green
$ns at 1.0 "$cbr0 start"
$ns at 2.0 "$cbr1 start"
$ns at 45 "finish"
$ns run
```

```
lan610
set ns [new Simulator]
set tf [open lab3.tr w]
$ns trace-all $tf
set nf [open lab3.nam w]
$ns namtrace-all $nf
$ns color 0 blue
set n0 [$ns node]
$n0 color "red"
set n1 [$ns node]
$n1 color "red"
set n2 [$ns node]
$n2 color "red"
set n3 [$ns node]
$n3 color "red"
set n4 [$ns node]
$n4 color "purple"
set n5 [$ns node]
$n5 color "purple"
set n6 [$ns node]
$n6 color "purple"
set n7 [$ns node]
$n7 color "purple"
$n1 label "Source/UDP"
$n3 label "Error node"
$n7 label "Destination"
set lan [$ns newLan "$n0 $n1 $n2 $n3" 100Mb 300ms LL
Queue/DropTail Mac/802_3]
set lan [$ns newLan "$n4 $n5 $n6 $n7" 100Mb 300ms LL
Queue/DropTail Mac/802_3]
$ns duplex-link $n3 $n4 100Mb 300ms DropTail
$ns duplex-link-op $n3 $n4 color "green"
set err [new ErrorModel]
$ns lossmodel $err $n3 $n4
$err set rate_ 0.1
set udp [new Agent/UDP]
$ns attach-agent $n1 $udp
set cbr [new Application/Traffic/CBR]
$cbr attach-agent $udp
$cbr set fid_ 0
$cbr set packetSize_ 1000
$cbr set interval_ 0.0001
set null [new Agent/Null]
$ns attach-agent $n7 $null
$ns connect $udp $null
proc finish { } {
global ns nf tf
$ns flush-trace
close $nf
close $tf
exec nam lab3.nam &
exit 0
}
$ns at 0.1 "$cbr start"
$ns at 2.0 "$cbr stop"
$ns at 3.0 "finish"
$ns run
File : lab3.awk
begin { pkt = 0;
time = 0;
udpacket = 0; }
{
 if($1=="r" && $3=="3" && $4=="4" && $5=="cbr")
{ pkt = pkt+$6;  time=$2; udpacket++; }
END {
printf "througput : %f MBPS\n",((pkt / time)*(8/1000000));
 printf "the packets received is : %d\n",udpacket;
printf "the packets calculated is : %d\n",pkt;
printf "the time duration is : %d\n",time; }
```

```
lannnodegraph
set ns [new Simulator]
set tf [open lab4.tr w]
$ns trace-all $tf
set nf [open lab4.nam w]
$ns namtrace-all $nf
$ns color 1 "blue"
$ns color 2 "green"
set n0 [$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"
$n5 label "dest1"
set lan [$ns newLan "$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
$tcp0 set fid_ 1
$tcp2 set fid_ 2
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 lab4.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:
begin { }
{ if ($6=="cwnd_")
printf ("%f\t%f\t\n", $1, $7);  }
end {}
Command:
-f awk filename.awk filename1.tr > a1
-f awk filename.awk filename2.tr > a2
To get Graph:
xgraph a1 a2
```

```
ess
set ns [new Simulator]
set tf [open lab8.tr w]
$ns trace-all $tf
set topo [new Topography]
$topo load_flatgrid 1000 1000
set nf [open lab8.nam w]
$ns namtrace-all-wireless $nf 1000 1000
$ns node-config -adhocRouting AODV \
                -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 lab8.nam  &
        close $tf
        exit 0
}
$ns at 250 "finish"
$ns run
awk File
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("Throughput from n0 to n1:%f Mbps\n",
        ((count1*pack1*8)/(time1*1000000)));
      printf("Throughput from n1 to n2:%f Mbps\n",
        ((count2*pack2*8)/(time2*1000000)));
}
```

```
pointopoint4node
set ns [ new Simulator ]
set tf [ open lab1.tr w ]
$ns trace-all $tf
set nf [ open lab1.nam w ]
$ns namtrace-all $nf
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
$ns color 1 "red"
$ns color 2 "blue"
$n0 label "Source/udp0"
$n1 label "Source/udp1"
$n2 label "Router"
$n3 label "Destination/Null"
$ns duplex-link $n0 $n2 10Mb 300ms
DropTail
$ns duplex-link $n1 $n2 10Mb 300ms
DropTail
$ns duplex-link $n2 $n3 1Mb 300ms
DropTail
$ns set queue-limit $n0 $n2 10
$ns set queue-limit $n1 $n2 10
$ns set queue-limit $n2 $n3 5
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
set null3 [new Agent/Null]
$ns attach-agent $n3 $null3
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
$udp0 set class_ 1
$udp1 set class_ 2
$ns connect $udp0 $null3
$ns connect $udp1 $null3
$cbr1 set packetSize_ 500
$cbr1 set interval_ 0.005
proc finish { } {
global ns nf tf
$ns flush-trace
exec nam lab1.nam &
close $tf
close $nf
exit 0
}
#Schedule the Events
$ns at 0.1 "$cbr0 start"
$ns at 0.1 "$cbr1 start"
$ns at 10.0 "finish"
$ns run
File: lab1.awk
#!/usr/bin/awk -f
BEGIN{ cbrPkt=0;
tcpPkt=0;}
 { if(($1 == "d")&&($5 == "cbr")) {
cbrPkt = cbrPkt + 1;  }
if(($1 == "d")&&($5 == "tcp")) {
tcpPkt = tcpPkt + 1; } }
END {
printf "\nNo. of CBR Packets
Dropped %d", cbrPkt;
printf "\nNo. of TCP Packets
Dropped %d", tcpPkt;
}
```

```
slide
#include<stdio.h>
int main() {
int w,i,f,frames[50];
printf("Enter window size: ");
scanf("%d",&w);
printf("\nEnter number of frames to
transmit: ");
scanf("%d",&f);
printf("\nEnter %d frames: ",f);
for(i=1;i<=f;i++)
 scanf("%d",&frames[i]);
printf("\nWith sliding window protocol
the frames will be sent in the following
manner (assuming no corruption of
frames)\n\n");
   printf("After sending %d frames at each
stage sender waits for acknowledgement
sent by the receiver\n\n",w);
    for(i=1;i<=f;i++) {
      if(i%w==0)
      {
         printf("%d\n",frames[i]);
         printf("Acknowledgement of above
frames sent is received by sender\n\n");
      }
      else
         printf("%d ",frames[i]);
   }
    if(f%w!=0)
      printf("\nAcknowledgement of above
frames sent is received by sender\n");
    return 0; }
```

```
distance vector
#include<stdio.h>
struct node
{ intdist[20];
int from[20];
} router[10];
int main() {
intdmat[20][20];
intno,i,j,k,count=1;
printf("\nEnter the number of nodes :\n");
scanf("%d",&no);
printf("\nEnter the distance matrix :\n");
for(i=1;i<=no;i++)
for(j=1;j<=no;j++) {
scanf("%d",&dmat[i][j]);
dmat[i][i]=0;
router[i].dist[j]=dmat[i][j];
router[i].from[j]=j; }
do {
for(i=1;i<=no;i++)
for(j=1;j<=no;j++)
for(k=1;k<=no;k++)
if(router[i].dist[j]>dmat[i][k]+router[k].dist[j])  {
router[i].dist[j]=router[i].dist[k]+router[k].dist[j];
router[i].from[j]=k;  }
count++;
}while(count<no);
for(i=1;i<=no;i++) {
printf("\nRouter table for router %c is \n",i+64);
for(j=1;j<=no;j++)
printf("\tNode %d Via %d, Distance : %d\n",j,router[i].from[j],router[i].dist[j]); }
return 0; }
```

```
tcp_udp
set ns [new Simulator]
set tf [open lab2.tr w]
$ns trace-all $tf
set nf [open lab2.nam w]
$ns namtrace-all $nf
set n0 [$ns node] set n1 [$ns node]
set n2 [$ns node] set n3 [$ns node]
$ns color 1 "red" $ns color 2 "blue"
$n0 label "source/TCP" $n1 label "source/UDP"
$n2 label "Router" $n3 label "destination"
$ns duplex-link $n0 $n2 100Mb 1ms DropTail
$ns duplex-link $n1 $n2 100Mb 1ms DropTail
$ns duplex-link $n2 $n3 100Mb 1ms DropTail
$ns duplex-link-op $n0 $n2 color "green"
$ns duplex-link-op $n0 $n2 label "from 0-2"
$ns duplex-link-op $n1 $n2 color "green"
$ns duplex-link-op $n1 $n2 label "from 1-2"
$ns duplex-link-op $n2 $n3 color "green"
$ns duplex-link-op $n2 $n3 label "from 2-3"
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set ftp0 [new Application/FTP]
$ftp0 attach-agent $tcp0
set sink3 [new Agent/TCPSink]
$ns attach-agent $n3 $sink3
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
set null3 [new Agent/Null]
$ns attach-agent $n3 $null3
$ftp0 set packetSize_ 500
$ftp0 set interval_ 0.001
$cbr1 set packetSize_ 500
$cbr1 set interval_ 0.001
$tcp0 set class_ 1
$udp1 set class_ 2
$ns connect $tcp0 $sink3
$ns connect $udp1 $null3
proc finish { } {
global ns nf tf
$ns flush-trace
exec nam lab2.nam &
close $nf
close $tf
exit 0 }
$ns at 0.1 "$cbr1 start"
$ns at 0.2 "$ftp0 start"
$ns at 5.0 "finish"
$ns run
BEGIN {  TCPSend=0; CBRSend=0; TCPDrop=0; CBRDrop=0; TCPDropRatio=0.0;
UDPDropRatio=0.0; TCPArrivalRatio=0.0;
                CBRArrivalRatio=0.0; }
{  src=$3; des=$4; type=$5;
event=$1;
if((src=="0")&&(des=="2")&&(event=="+")) { TCPSend++; }
if((src=="1")&&(des=="2")&&(event=="+")) { CBRSend++; }
if((event=="d")&&(type=="tcp")) { TCPDrop++;  }
if((event=="d")&&(type=="cbr")) {  CBRDrop++; }
}END {
printf "\nTCPSend %d", TCPSend; printf "\nCBRSend %d", CBRSend;
printf "\nTCPDrop %d", TCPDrop; printf "\nCBRDrop %d", CBRDrop;
TCPArrivalRatio=((TCPSend-TCPDrop)/TCPSend);
TCPDropRatio=(TCPDrop/TCPSend);
UDPArrivalRatio=((CBRSend-CBRDrop)/CBRSend);
UDPDropRatio=(CBRDrop/CBRSend);
printf "\nTCPArrivalRatio %f", TCPArrivalRatio;
printf "\nTCPDropRatio %f", TCPDropRatio;
printf "\nUDPArrivalRatio %f", UDPArrivalRatio;
printf "\nUDPDropRatio %f", UDPDropRatio; }
```