Table of Contents

17ECL68 : COMPUTER COMMUNICATION NETWORK LAB........................1
A. LABORATORY INFORMATION........................1
1. Lab Overview........................1
2. Lab Content........................2
3. Lab Material........................2
4. Lab Prerequisites:........................3
5. General Instructions........................3
6. Lab Specific Instructions........................3
B. OBE PARAMETERS........................4
1. Lab / Course Outcomes........................4
2. Lab Applications........................4
3. Articulation Matrix........................4
4. Mapping Justification........................5
5. Curricular Gap and Content........................6
6. Content Beyond Syllabus........................6
C. COURSE ASSESSMENT........................6
1. Course Coverage........................6
2. Continuous Internal Assessment (CIA)........................7
D. EXPERIMENTS........................8
Experiment 01 : TCP/TCP transmission protocol........................8
Experiment 02 : TCP/UDP Transmission protocol........................11
Experiment 03 : Ethernet LAN throughput calculation........................17
Experiment 04 : Ethernet LAN with N Nodes........................19
Experiment 05 : Wireless LAN........................22
Experiment 06 : Link state routing algorithm........................25
Experiment 07 : HDLC frame to perform Bit and character stuffing........................29
Experiment 08 : Distance vector algorithm to find suitable path for transmission........................32
Experiment 09 : Dijkstra's Algorithm to compute shortest path........................36
Experiment 10 : Using CRC-CCITT polynomial to obtain CRC code........................41
Experiment 11 : Implementation of Stop & Wait protocol........................45
Experiment 12 : Congesion control using Leakg bucket algorithm........................47
VIVA QUESTIONS........................50

Note : Remove "Table of Content" before including in CP Book

17ECL68 : COMPUTER COMMUNICATION NETWORK LAB

A. LABORATORY INFORMATION

1. Lab Overview

Degree:	BE	*Program:*	EC
Year / Semester :	3/6	*Academic Year:*	2019-20
Course Title:	Computer Communication Network Lab	*Course Code:*	17ECL68
Credit / L-T-P:	4 / 4-0-0	*SEE Duration:*	180 Minutes
Total Contact Hours:	62 Hrs	*SEE Marks:*	80 Marks
CIA Marks:	20	*Assignment*	
Course Plan Author:	Kiranmayi , N S MYTHREYE,	*Sign*	Dt :
Checked By:		*Sign*	Dt :

2. Lab Content

Unit	Title of the Experiments	Lab Hours	Concept	Blooms Level
1	Implimentation of Point to Point network with 4 nodes	3	Point to Point network with 4 nodes	L4
2	Implimentation of Point to Point network appling TCP & UDP agent	3	Point to Point network appling TCP & UDP agent	L4
3	Implimentation of Ethernet LAN to compare error rate & data rate	3	Ethernet LAN to compare error rate & data rate	L4
4	Implimentation of Ethernet LAN to Obtain congestion window	3	Ethernet LAN to Obtain congestion window	L4
5	Implimentation of Wireless LAN	3	Wireless LAN	L4
6	Implimentation of Link state routing algorithm	3	Link state routing algorithm	L4
7	HDLC frame to perform Bit stuffing	3	Bit stuffing	L4
8	Distance vector algorithm to find suitable path for transmission	3	Shortest path algorithm	L4
9	Dijkstra's Algorithm to compute shortest path	3	Dijkstra's Algorithm	L4
10	Using CRC-CCITT polynomial to obtain CRC code	3	CRC-CCITT polynomial	L4
11	Implementation of Stop & Wait protocol	3	Stop & Wait protocol	L4
12	Congesion control using Leakg bucket algorithm	3	Leaky bucket	

3. Lab Material

Unit	Details	Available
1	Text books	
	Data Communications and Networking , Forouzan, 5th Edition, McGraw Hill, 2016 ISBN: 1-25-906475-3	In Lib
2	Reference books	
	1. Computer Networks, James J Kurose, Keith W Ross, Pearson Education, 2013, ISBN: 0-273-76896-4	In dept
	2. Introduction to Data Communication and Networking, Wayarles Tomasi, Pearson Education, 2007, ISBN:0130138282	
3	Others (Web, Video, Simulation, Notes etc.)	
		Not Available

4. Lab Prerequisites:

-	-	Base Course:		-	-
SNo	Course Code	Course Name	Topic / Description	Sem	Remarks
1	17EC61	Digital communication	Knowledge on Multiplexing	6	
2	17EC61	Digital communication	Knowledge on Digital modulation techniques	6	
3	17EC71	Microwaves and Antennas	Knowledge on microwave active and passive components and antennas	7	Plan Gap Course
4	17EC82	Fiber Optics and Networks	Knowledge on optical fibers and antennas	8	Plan Gap Course
5	17EC61	Digital communication	Knowledge NRZ, RZ binary polar signalling	6	

Note: If prerequisites are not taught earlier, GAP in curriculum needs to be addressed. Include in Remarks and implement in B.5.

5. General Instructions

SNo	Instructions	Remarks
1	Observation book and Lab record are compulsory.	
2	Students should report to the concerned lab as per the time table.	
3	After completion of the program, certification of the concerned staff in-charge in the observation book is necessary.	
4	Student should bring a notebook of 100 pages and should enter the readings /observations into the notebook while performing the experiment.	
5	The record of observations along with the detailed experimental procedure of the experiment in the Immediate last session should be submitted and certified staff member in-charge.	
6	Should attempt all problems / assignments given in the list session wise.	
7	It is responsibility to conduct the experiment individually.	
8	When the experiment is completed, the student should disconnect the setup made by them, and should return all the components/instruments taken for the purpose.	
9	Any damage of the equipment or burn-out components will be viewed seriously either by putting penalty or by dismissing the total group of students from the lab for the semester/year	
10	Completed lab assignments should be submitted in the form of a Lab Record in which you have to write the Aim, components required, theory, procedure, circuit diagram and design along with graphs/ waveform and results for given design values	

6. Lab Specific Instructions

SNo	Specific Instructions	Remarks
1	Rules established in lecture/lab regarding protection, working with exposed high voltage, horseplay, etc. apply to all individuals working in the lab area.	
2	Working alone in the lab will not be permitted where exposed voltages exceeding 25 volts are present.	
3	Carry out the experiments in such a way that the equipment will not be damaged or destroyed.	

4	Follow all written and verbal instructions carefully. If you do not understand the instructions, the handouts and the procedures, ask the instructor or teaching assistant	
5	The workplace has to be tidy before, during and after the experiment.	
6	Read the handout and procedures before starting the experiments	
7	Turn off all test equipment, return equipment, tools, and test leads to their proper storage area	

B. OBE PARAMETERS

1. Lab / Course Outcomes

#	COs	Teach. Hours	Concept	Instr Method	Assessment Method	Blooms' Level
1	Choose suitable tool to model a network and understand the protocols at various OSI reference levels.	9	Point to point network	Simulation	Assignment and Slip Test	L4
2	Design a suitable network and simulate using a Network simulator tool.	9	LAN and link state algorithm	Simulation	Assignment and Slip Test	L4
3	Simulate the networking concepts and protocols using C/C++.	9	Framing and routing	Simulation	Assignment and Slip Test	L4
4	Model the networks for different configuration and analyze the results.	9	Coding and congestion control	Simulation	Assignment and Slip Test	L4
-	**Total**	**36**	-	-	-	-

Note: Identify a max of 2 Concepts per unit. Write 1 CO per concept.

2. Lab Applications

SNo	Application Area	CO	Level
1	Evaluate time and space complexity and calculate performance	CO1	L4
2	Understanding searching and sorting	CO2	L4
3	Use AND / OR graph, spanning trees	CO3	L4
4	Use Backtracking technique for searching a set of solutions or for searching an optimal solution	CO4	L4
5	Apply Greedy method for finding optimal solution	CO5	L4
6	Apply Dynamic Programming to find a sequence of decisions	CO6	L4
7	Evaluate traveling sales man problem by using dynamic programming	CO7	L4
8	Apply Branch and Bound for solving combinatorial optimization problems	CO8	L2
9	Able to differentiate NP – Hard and NP – Complete Problems	CO9	L2

Note: Write 1 or 2 applications per CO.

3. Articulation Matrix

(CO – PO MAPPING)

-	Course Outcomes	Program Outcomes												
#	COs	PO 1	PO 2	PO 3	PO 4	PO 5	PO 6	PO 7	PO 8	PO 9	PO 10	PO 11	PO 12	Level
17ECL68.1	Choose suitable tool to model a network and understand the	3	3	3	3	-	-	-	-	3	-	-	-	L4

	protocols at various OSI reference levels.													
17ECL68.2	Design a suitable network and simulate using a Network simulator tool.	3	3	3	3	-	-	-	-	3	-	-	-	L4
17ECL68.3	Simulate the networking concepts and protocols using C/C++.	3	3	3	3	-	-	-	-	3	-	-	-	L4
17ECL68.4	Model the networks for different configuration and analyze the results.	3	3	3	3	-	-	-	-	3	-	-	-	L4
17ECL68	Average	**3**	**3**	**3**	**3**	**-**	**-**	**-**	**-**	**3**	**-**	**-**	**-**	

Note: Mention the mapping strength as 1, 2, or 3

4. Mapping Justification

Mapping		Mapping Level	Justification
CO	**PO**	**-**	**-**
CO1	PO1		The experiment has the application of mathematics, science and engineering fundamentals
CO1	PO2		The conduction of this experiment includes the identification, formulation and analysis, reaching the substantiated conclusions
CO1	PO3		Design and development of the circuit is involved in this experiment
CO1	PO4		The problems faced during the design and conduction of the experiment is investigated and resolved
CO1	PO5		Modern tool NS2 and NSG2.1 are used in the simulation process
CO1	PO9		The experiments are conducted in multidisciplinary functional teams
CO2	PO1		The experiment has the application of mathematics, science and engineering fundamentals
CO2	PO2		The conduction of this experiment includes the identification, formulation and analysis, reaching the substantiated conclusions
CO2	PO3		Design and development of the circuit is involved in this experiment
CO2	PO4		The problems faced during the design and conduction of the experiment is investigated and resolved
CO1	PO5		Modern tool NS2 and NSG2.1 are used in the simulation process
CO2	PO9		The experiments are conducted in multidisciplinary functional teams
CO3	PO1		The experiment has the application of mathematics, science and engineering fundamentals
CO3	PO2		The conduction of this experiment includes the identification, formulation and analysis, reaching the substantiated conclusions
CO3	PO3		Design and development of the circuit is involved in this experiment
CO3	PO4		The problems faced during the design and conduction of the experiment is investigated and resolved
CO3	PO9		The experiments are conducted in multidisciplinary functional teams
CO4	PO1		The experiment has the application of mathematics, science and engineering fundamentals
CO4	PO2		The conduction of this experiment includes the identification, formulation and analysis, reaching the substantiated conclusions
CO4	PO3		Design and development of the circuit is involved in this experiment
CO4	PO4		The problems faced during the design and conduction of the experiment is

			investigated and resolved
CO4	PO9		The experiments are conducted in multidisciplinary functional teams

Note: Write justification for each CO-PO mapping.

5. Curricular Gap and Content

SNo	Gap Topic	Actions Planned	Schedule Planned	Resources Person	PO Mapping
1	Computer communication networks	Theoritical introduction in the class and demonstration in lab		Self	

Note: Write Gap topics from A.4 and add others also.

6. Content Beyond Syllabus

SNo	Gap Topic	Actions Planned	Schedule Planned	Resources Person	PO Mapping
1					
2					
3					
4					
5					
6					
7					

Note: Anything not covered above is included here.

C. COURSE ASSESSMENT

1. Course Coverage

Unit	Title	Teaching Hours	No. of question in Exam							CO	Levels
			CIA-1	CIA-2	CIA-3	Asg-1	Asg-2	Asg-3	SEE		
1	Implement a point to point network with four nodes and duplex links between them. Analyze the network performance by setting the queue size and varying bandwidth	3	1	-	-	-	-	-	1	CO1	L4
2	Implement a four node point to point network with link n0-n2, n1-n2 and n2-n3. Apply Tcp agent between n0-n3 and UDP between n1-n3. Apply relevant applications over TCP and UDP agents changing the parameters and determine the number of packets sent by TCP/UDP	3	1	-	-	-	-	-	1	CO1	L4
3	Implement Ethernet LAN using n (6-10) nodes. Compare the throughput by changing the error rate and data rate.	3	-	1	-	-	-	-	1	CO1	L4
4	Implement ethernet LAN using n	3	-	1	-	-	-	-	1	CO2	L4

	nodes and assign multiple traffic to the nodes and obtain congestion window for diffferent source/destinations										
5	Implement ESS with transmission nodes in wireless LAN and obtain the performance parameters	3	-	-	1	-	-	-	1	CO2	L4
6	Implementation of the link state routing algorithm	3	-	-	1	-	-	-	1	CO2	L4
7	Write a program for the HDLC frame to perform Bit stuffing and character stuffing	3	1	-	-	-	-	-	1	CO3	L4
8	Write a program for distance vector algorithm to find the suitable path for transmission	3	1	-	-	-	-	-	1	CO3	L4
9	Implement Dijkstras' algorithm to compute the shortest routing path	3	-	1	-	-	-	-	1	CO3	L4
10	For the given data, use CRC-CCITT polynomial to obtain CRC code. Verify the program for the cases with and without error	3	-	1	-	-	-	-	1	CO4	L4
11	Implementation od stop and wait protocol and sliding window protocol	3	-	-	1	-	-	-	1	CO4	L4
12	Write a program for congestion control using leaky bucket algorthm	3	-	-	1	-	-	-	1	CO4	L4
-	**Total**	**36**	**4**	**4**	**4**	-	-	-		-	-

Note: Write CO based on the theory course.

2. Continuous Internal Assessment (CIA)

Evaluation	Weightage in Marks	CO	Levels
CIA Exam – 1	12	CO1, CO3,	L4
CIA Exam – 2	12	CO1, CO2,CO3,CO4	L4
CIA Exam – 3	12	CO3,CO4	L4
Assignment - 1	8	CO1, CO3,	L4
Assignment - 2	8	CO1, CO2,CO3,CO4	L4
Assignment - 3	8	CO3,CO4	L4
Seminar - 1			
Seminar - 2			
Seminar - 3			
Other Activities – define – Slip test			
Final CIA Marks	**20**	-	-

-

SNo	**Description**	**Marks**
1	Observation and Weekly Laboratory Activities	04 Marks
2	Record Writing	8 Marks for each Expt

3	Internal Exam Assessment	8 Marks
4	Total Internal Assessment	20 Marks
5	SEE	80 Marks
-	**Total**	**100 Marks**

D. EXPERIMENTS

Experiment 01 : TCP/TCP transmission protocol

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Implement a point to point network with four nodes and duplex links between them.						
2	Course Outcomes	Choose suitable tool to model a network and understand the protocols at various OSI reference levels.						
3	Aim	Implement a point to point network with four nodes and duplex links between them. Analyze the network performance by setting the queue size and varying the bandwidth.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	TCP (Transmission Control protocol) is a standard that defines how to establish and maintain a network conversation via which application programs can exchange data. TCP works with the Internet Protocol (IP), which defines how computers send packetss of data to each other. Together, TCP and IP are the basic rules defining the Internet.						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	#================================= # 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 lab1.tr w]						

```
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open lab1.nam w]
$ns namtrace-all $namfile

#===================================
#       Nodes Definition
#===================================
#Create 4 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#==========================================
# These below line helps to define color, label and shape
#==========================================

$ns at 0.0 "$n0 color blue"
$ns at 0.0 "$n1 color red"

$ns at 0.0 "$n0 label Source/Tcp0"
$ns at 0.0 "$n1 label Source/Tcp1"
$ns at 0.0 "$n2 label Router"
$ns at 0.0 "$n3 label Destination"

$n0 shape hexagon
$n1 shape hexagon
$n3 shape square

#===================================
#       Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n0 $n2 100.0Mb 20ms DropTail
$ns queue-limit $n0 $n2 20                          #Q1
$ns duplex-link $n1 $n2 100.0Mb 20ms DropTail
$ns queue-limit $n1 $n2 20                        #Q2
$ns duplex-link $n2 $n3 100.0Mb 20ms DropTail
$ns queue-limit $n2 $n3 20                        #Q3

#Above Highlighted Bandwidth and Queue Size will be chang

#Give node position (for NAM)
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n1 $n2 orient right-up
$ns duplex-link-op $n2 $n3 orient right

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

```
$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 $n3 $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 $tcp0
$ns at 1.0 "$ftp0 start"
$ns at 8.0 "$ftp0 stop"

#Setup a FTP Application over TCP connection
set ftp1 [new Application/FTP]
$ftp1 attach-agent $tcp1
$ns at 1.0 "$ftp1 start"
$ns at 8.0 "$ftp1 stop"

#===================================
#       Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam lab1.nam &
    exit 0
}
$ns at $val(stop) "$ns nam-end-wireless $val(stop)"
```

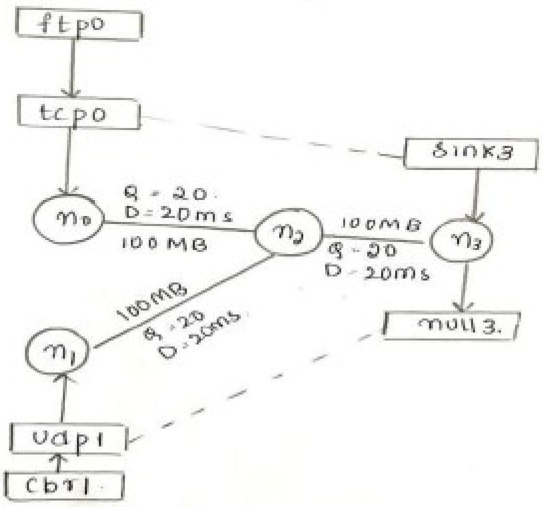
```
$ns run


Awk Script:

BEGIN{
tcppack=0
tcppack1=0
}
{
if($1=="r"&&$4=="3"&&$5=="tcp"&&$6=="1540")
{
tcppack++;
}
if($1=="d"&&$3=="2"&&$5=="tcp"&&$6=="1540")
{
tcppack1++;
}

}
END{
printf("\n total number of data packets received at Node 3: %d\n", tcppack++);
printf("\n total number of packets dropped at Node 2: %d\n", tcppack1++);
}
```

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	see table below
11	Results & Analysis	
12	Application Areas	FTP, data and control), which is used in sending large files. Simple Mail Transfer Protocol (SMTP)
13	Remarks	
14	Faculty Signature with Date	

Bandwidth(MB) N0-N2, N1-N2, N2-N3	Queue Size Q1, Q2, Q3	Received at Node 3	Dropped at Node 2
100,100,100	20, 20,20	6820	0
100,100,1	20, 20,2	508	45
300,300,10	50, 50,3	3021	48

Experiment 02 : TCP/UDP Transmission protocol

-	Experiment No.:	2	Marks		Date Planned		Date Conducted	
1	Title	Implement a four node point to point network with links n0-n2, n1-n2 and						

		n2-n3. 1. Apply TCP agent between n0-n3 and UDP between n1-n3. Apply relevant 2. applications over TCP and UDP agents changing the parameter and determine the 3. number of packets sent by TCP/UDP.
2	Course Outcomes	Choose suitable tool to model a network and understand the protocols at various OSI reference levels.
3	Aim	Implement a four node point to point network with links n0-n2, n1-n2 and n2-n3. 4. Apply TCP agent between n0-n3 and UDP between n1-n3. Apply relevant 5. applications over TCP and UDP agents changing the parameter and determine the 6. number of packets sent by TCP/UDP.
4	Material / Equipment Required	Lab Manual
5	Theory, Formula, Principle, Concept	UDP (User Datagram Protocol) is an alternative communications protocol to Transmission ControlProtocol (TCP) used primarily for establishing low-latency and loss-tolerating connections between applications on the internet.
6	Procedure, Program, Activity, Algorithm, Pseudo Code	
7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	

8	Observation Table, Look-up Table, Output	(code below)

```
#===================================
#    Simulation parameters setup
#===================================
set val(stop)   50.0                    ;# time of simulation end

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

#Open the NS trace file
set tracefile [open lab2.tr w]
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open lab2.nam w]
$ns namtrace-all $namfile

#===================================
#       Nodes Definition
#===================================
#Create 4 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]

#=====================================================
# These below line helps to define color, label and shape
#=====================================================

$ns color 1 "blue"
$ns color 2 "red"

$ns at 0.0 "$n0 color blue"
$ns at 0.0 "$n1 color red"

$ns at 0.0 "$n0 label Source/Tcp0"
```

```
$ns at 0.0 "$n2 label Router"
$ns at 0.0 "$n3 label Destination"

$n0 shape hexagon
$n1 shape hexagon
$n3 shape square

#===================================
#       Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n0 $n2 200.0Mb 10ms DropTail
$ns queue-limit $n0 $n2 50
$ns duplex-link $n2 $n3 200.0Mb 10ms DropTail
$ns queue-limit $n2 $n3 50
$ns duplex-link $n1 $n2 200.0Mb 10ms DropTail
$ns queue-limit $n1 $n2 50

#Give node position (for NAM)

$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n2 $n3 orient right
$ns duplex-link-op $n1 $n2 orient right-up

$ns duplex-link-op $n2 $n3 color "green"
$ns duplex-link-op $n0 $n2 label "TCP Packets"
$ns duplex-link-op $n1 $n2 label "UDP Packets"
$ns duplex-link-op $n2 $n3 label "UDP+TCP Packets"



#===================================
#       Agents Definition
#===================================
#Setup a TCP connection
set tcp0 [new Agent/TCP]
$ns attach-agent $n0 $tcp0
set sink3 [new Agent/TCPSink]
$ns attach-agent $n3 $sink3
$ns connect $tcp0 $sink3
```

```
$tcp0 set interval_ 0.1

#Setup a UDP connection
set udp1 [new Agent/UDP]
$ns attach-agent $n1 $udp1
set null2 [new Agent/Null]
$ns attach-agent $n3 $null2
$ns connect $udp1 $null2
$udp1 set packetSize_ 1500
$udp1 set interval_ 0.1

$tcp0 set class_ 1
$udp1 set class_ 2

#===================================
#       Applications Definition
#===================================
#Setup a FTP Application over TCP connection
set ftp0 [new Application/FTP]

$ftp0 attach-agent $tcp0
$ns at 1.0 "$ftp0 start"
$ns at 9.0 "$ftp0 stop"

#Setup a CBR Application over UDP connection
set cbr1 [new Application/Traffic/CBR]
$cbr1 attach-agent $udp1
$cbr1 set packetSize_ 1000    # Change the Packet Size
$cbr1 set rate_ 1.0Mb
$cbr1 set random_ null
$ns at 1.0 "$cbr1 start"
$ns at 9.0 "$cbr1 stop"


#===================================
#       Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
```

```
    close $tracefile
    close $namfile
    exec nam lab2.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 Script:

```
 BEGIN{
 tcppack=0
 tcppack1=0
 }
 {
 if($1=="r"&&$4=="2"&&$5=="tcp"&&$6=="1540") #1540, 1040, 1340
 {
tcppack++;
}
if($1=="r"&&$4=="2"&&$5=="cbr"&&$6=="1500") #1500, 1500, 1500
{
tcppack1++;
}

}
END{
printf("\n total number of TCP data packets sent between Node 0 and Node 2:
%d\n", tcppack++);
printf("\n total number of UDP data packets sent between Node 1 and Node 2:
%d\n", tcppack1++);
}
```

9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	(see table below)
12	Application Areas	low-latency and loss-tolerating connections betweenapplications on the internet.
13	Remarks	
14	Faculty Signature with Date	

Simulation Time (Sec)	Packet Size (Byte)	Sent at Node 2
Start Time:1 Stop Time:9	TCP:1500 UDP:1500	TCP:3930 UDP:1001
Start Time:1 Stop Time:20	TCP:1000 UDP:1500	TCP:9410 UDP:1584
Start Time:1 Stop Time:25	TCP:1300 UDP:1500	TCP:11910 UDP:2000

Experiment 03 : Ethernet LAN throughput calculation

-	Experiment No.:	1	Marks		Date Planned		Date Conducted	
1	Title	Implement Ethernet LAN using n (6-10) nodes. Compare the throughput by changing the error rate and data rate.						
2	Course Outcomes	Choose suitable tool to model a network and understand the protocols at various OSI reference levels.						
3	Aim	Implement Ethernet LAN using n (6-10) nodes. Compare the throughput by changing the error rate and data rate.						
4	Material / Equipment Required							
5	Theory, Formula, Principle, Concept	ETHERNET. Local Area Network (LAN) is a data communication networkconnecting various terminals or computers within a building or limited geographical area.The connection among the devices could wired or wireless. Ethernet, Token Ring and Wireless LAN using IEEE 802.11 are examples of standard LANtechnologies.						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	(see code below)						

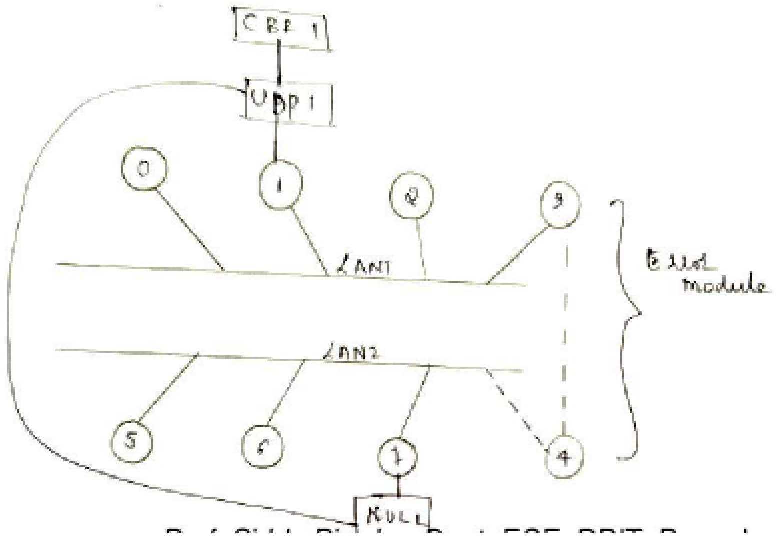
```
set ns [new Simulator]
set tf [open lab3.tr w]
$ns trace-all $tf

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]
$n4 color "magenta"
set n5 [$ns node]
$n5 color "magenta"
set n6 [$ns node]
$n6 color "magenta"
set n7 [$ns node]
$n7 color "magenta"

$n1 label "Source/UDP"
$n3 label "Error Node"
$n7 label "Destination/Null"

$ns make-lan "$n0 $n1 $n2 $n3" 100Mb 300ms LL Queue/DropTail Mac/802_3
$ns make-lan "$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"
```

```
# lossmodel is a command and it is single word. Space should not be given
between loss and model

set err [new ErrorModel]
$ns lossmodel $err $n3 $n4
$err set rate_ 0.1                  #Change the Error Rate 0.1, 0.3, 0.5,

# error rate should be changed for each output like 0.1,0.3,0.5.... */
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_ 0.001            #Change the Data Rate 0.001, 0.01, 0.1,
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 3.0 "finish"
$ns run
```

Awk Script:

```
BEGIN{
tcppack=0
tcppack1=0
}
{
if($1=="r"&&$4=="7"&&$5=="cbr"&&$6=="1000")
{
tcppack++;
}
}
END{
printf("\n total number of  data packets at Node 7: %d\n", tcppack++);
}
```

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	

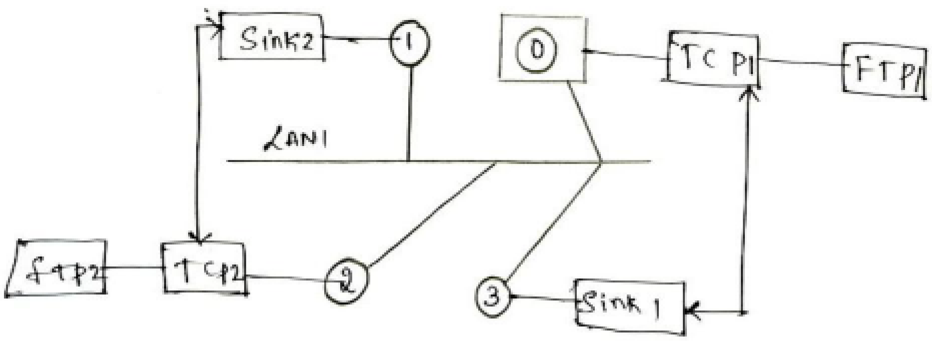
<table>
<tr><td></td><td></td><td>
</td></tr>
<tr><td>8</td><td>Observation Table, Look-up Table, Output</td><td></td></tr>
<tr><td>9</td><td>Sample Calculations</td><td></td></tr>
<tr><td>10</td><td>Graphs, Outputs</td><td></td></tr>
<tr><td>11</td><td>Results & Analysis</td><td>
<table>
<tr><th>Error Rate</th><th>Data Rate</th><th>Received at Node 7</th></tr>
<tr><td>0.1</td><td>0.001</td><td>1255</td></tr>
<tr><td>0.3</td><td>0.01</td><td>95</td></tr>
<tr><td>0.5</td><td>0.1</td><td>05</td></tr>
</table>
</td></tr>
<tr><td>12</td><td>Application Areas</td><td>Provides better quality of service (QoS). Voice: Voice over Internet Protocol (VoIP) is well-known for its cost-savings</td></tr>
<tr><td>13</td><td>Remarks</td><td></td></tr>
<tr><td>14</td><td>Faculty Signature with Date</td><td></td></tr>
</table>

Experiment 04 : Ethernet LAN with N Nodes

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Implement Ethernet LAN using n nodes and assign multiple traffic to the nodes and obtain congestion window for different sources/ destinations.						
2	Course Outcomes	Design a suitable network and simulate using a Network simulator tool.						
3	Aim	Implement Ethernet LAN using n nodes and assign multiple traffic to the nodes and obtain congestion window for different sources/ destinations.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula,	Ethernet is the traditional technology for connecting wired local area						

	Principle, Concept	networks (LANs), enabling devices to communicate with each other via a protocol -- a set of rules or common network language.
6	Procedure, Program, Activity, Algorithm, Pseudo Code	set n0 [$ns node] set n1 [$ns node] set n2 [$ns node] set n3 [$ns node] $ns make-lan "$n0 $n1 $n2 $n3" 10mb 10ms LL Queue/DropTail Mac/802_3 $ns color 1 "blue" $ns color 2 "red" $ns at 0.0 "$n3 color blue" $ns at 0.0 "$n1 color red" $ns at 0.0 "$n0 label Source/Tcp0" $ns at 0.0 "$n2 label Source/Tcp2" $ns at 0.0 "$n1 label Destination/Sink1" $ns at 0.0 "$n3 label Destination/Sink3" $n0 shape hexagon $n2 shape hexagon $n1 shape square $n3 shape square set tcp0 [new Agent/TCP/Reno] $ns attach-agent $n0 $tcp0 set ftp0 [new Application/FTP] $ftp0 attach-agent $tcp0 set sink3 [new Agent/TCPSink] $ns attach-agent $n3 $sink3 $ns connect $tcp0 $sink3 set tcp2 [new Agent/TCP] $ns attach-agent $n2 $tcp2 set ftp2 [new Application/FTP] $ftp2 attach-agent $tcp2 set sink1 [new Agent/TCPSink] $ns attach-agent $n1 $sink1 $ns connect $tcp2 $sink1 $tcp0 set class_ 1 $tcp2 set class_2

```
######To trace the congestion window##########
set file1 [open file1.tr w]
$tcp0 attach $file1
$tcp0 trace cwnd_
#$tcp0 set maxcwnd_ 10
set file2 [open file2.tr w]
$tcp2 attach $file2
$tcp2 trace cwnd_
proc finish { } {
global nf tf ns
$ns flush-trace
exec nam lab4.nam &
close $nf
close $tf
exit 0
}
$ns at 0.1 "$ftp0 start"
$ns at 1.5 "$ftp0 stop"
$ns at 2 "$ftp0 start"

$ns at 3 "$ftp0 stop"
$ns at 0.2 "$ftp2 start"
$ns at 2 "$ftp2 stop"
$ns at 2.5 "$ftp2 start"
$ns at 4 "$ftp2 stop"
$ns at 5.0 "finish"

$ns run
```

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	

11	Results & Analysis	Time / TCP / TCPReno: 0.5 / 23.02 / 21.91; 1 / 29.73 / 29.53; 1.5 / 34.59 / 34.71; 2 / 2 / 41.11; 2.5 / 26.23 / 2; 3 / 32.84 / 24.62
12	Application Areas	Provides better quality of service (QoS). Voice: Voice over Internet Protocol (VoIP) is well-known for its cost-savings
13	Remarks	
14	Faculty Signature with Date	

Time	TCP	TCPReno
0.5	23.02	21.91
1	29.73	29.53
1.5	34.59	34.71
2	2	41.11
2.5	26.23	2
3	32.84	24.62

Experiment 05 : Wireless LAN

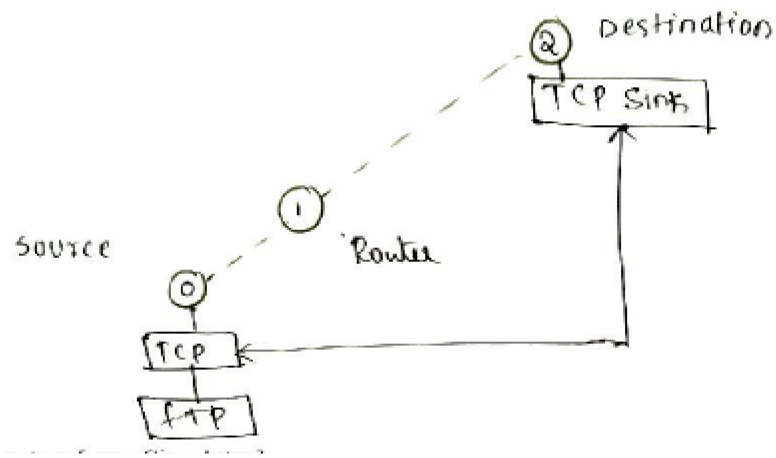
-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Implement ESS with transmission nodes in Wireless LAN and obtain the performance parameters.						
2	Course Outcomes	Design a suitable network and simulate using a Network simulator tool.						
3	Aim	Implement ESS with transmission nodes in Wireless LAN and obtain the performance parameters.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	As wireless LAN (WLAN) devices increase, the need to provide them connectivity extends to the outdoors.Applicationssuch as public wi-fi access, outdoor surveillance, and outdoor inventory control all stretch the need for outdoor wirelessaccess. Remote networks as well as client devices need to be connected.						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	(see code below)						

```
set nf [open lab5.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 OFF

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]
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 lab5.nam  &
        close $tf
        exit 0
}
$ns at 250 "finish"
$ns run

BEGIN{
tcppack=0
tcppack1=0
}
{
if($1=="s"&&$3=="_0_"&&$4=="AGT"&&$8=="1040")
{
tcppack++;
}
if($1=="r"&&$3=="_2_"&&$4=="AGT"&&$8=="1040")
{
tcppack1++;
}
}
}
END{
printf("\n total number of  data packets sent from Node 0: %d\n",
tcppack++);
printf("\n total number of  data packets received at Node 2: %d\n",
tcppack++);

}
```

Prof. Siddu Biradar, Dept. ECE, DBIT, Bangalore

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	

12	Application Areas	
13	Remarks	
14	Faculty Signature with Date	

Experiment 06 : Link state routing algorithm

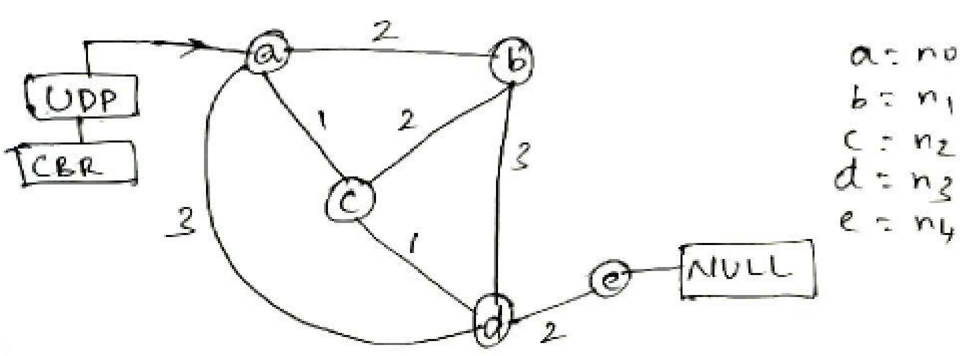
-	Experiment No.:	1	Marks		Date Planned		Date Conducted	
1	Title	Implementation of Link state routing algorithm.						
2	Course Outcomes	Design a suitable network and simulate using a Network simulator tool.						
3	Aim	Implementation of Link state routing algorithm.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	A routing method used by dynamic routers in which every router maintains a database of its individual autonomous system (AS) topology. The Open Shortest Path First (OSPF) routing protocol uses the link state routing algorithm to allow OSPFrouters to exchange routing information with each other.						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	(see below)						

```
#===================================
#    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 lab6.tr w]
$ns trace-all $tracefile

#Open the NAM trace file
set namfile [open lab6.nam w]
$ns namtrace-all $namfile
```

```
#Create 5 nodes
set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]
set n3 [$ns node]
set n4 [$ns node]

#===================================
#       Links Definition
#===================================
#Createlinks between nodes
$ns duplex-link $n0 $n1 100.0Mb 10ms DropTail
$ns queue-limit $n0 $n1 50
$ns duplex-link $n0 $n2 100.0Mb 10ms DropTail
$ns queue-limit $n0 $n2 50
$ns duplex-link $n2 $n3 100.0Mb 10ms DropTail
$ns queue-limit $n2 $n3 50
$ns duplex-link $n1 $n3 100.0Mb 10ms DropTail

$ns queue-limit $n1 $n3 50
$ns duplex-link $n3 $n4 100.0Mb 10ms DropTail
$ns queue-limit $n3 $n4 50
$ns duplex-link $n0 $n3 100.0Mb 10ms DropTail
$ns queue-limit $n0 $n3 50
$ns duplex-link $n1 $n2 100.0Mb 10ms DropTail
$ns queue-limit $n1 $n2 50

#Give node position (for NAM)
$ns duplex-link-op $n0 $n1 orient right
$ns duplex-link-op $n0 $n2 orient right-down
$ns duplex-link-op $n2 $n3 orient right
$ns duplex-link-op $n1 $n3 orient left-down
$ns duplex-link-op $n3 $n4 orient left-down
$ns duplex-link-op $n0 $n3 orient right-down
$ns duplex-link-op $n1 $n2 orient left-down

#Set the link costs. All link costs are symmetric
```

```
$ns cost $n0 $n3 3

$ns cost $n1 $n0 2
$ns cost $n1 $n2 2
$ns cost $n1 $n3 3

$ns cost $n2 $n1 2
$ns cost $n2 $n0 1
$ns cost $n2 $n3 1

$ns cost $n3 $n2 1
$ns cost $n3 $n1 3
$ns cost $n3 $n0 3
$ns cost $n3 $n4 2

$ns cost $n4 $n3 2

#===================================
#       Agents Definition
#===================================
#Setup a UDP connection
set udp0 [new Agent/UDP]
$ns attach-agent $n0 $udp0
set null1 [new Agent/Null]
$ns attach-agent $n4 $null1
$ns connect $udp0 $null1
$udp0 set packetSize_ 1500

#===================================
#       Applications Definition
#===================================
#Setup a CBR Application over UDP connection
set cbr0 [new Application/Traffic/CBR]
$cbr0 attach-agent $udp0
$cbr0 set packetSize_ 1000
$cbr0 set rate_ 1.0Mb
$cbr0 set random_ null
$ns at 1.0 "$cbr0 start"
$ns at 5.0 "$cbr0 stop"
```

```
$ns rtproto LS
#===================================
#       Termination
#===================================
#Define a 'finish' procedure
proc finish {} {
    global ns tracefile namfile
    $ns flush-trace
    close $tracefile
    close $namfile
    exec nam lab6.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
```

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	1. N0-N1-N2-N3-N4: Total Cost is of **7** 2. N0-N2- N1 -N3-N4 : Total Cost is of **8** 3. N0-N2-N3-N4 : Total Cost is of **4** 4. N0-N3-N4 : Total Cost is of **5** • **Shortest according to Link State algorithm** is **N0-N2-N3-N4** having Total Cost is of **4**
11	Results & Analysis	
12	Application Areas	routers to exchange routing information , distance vector routers
13	Remarks	

14	Faculty Signature with Date	

Experiment 07 : HDLC frame to perform Bit and character stuffing

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	HDLC frame to perform Bit stuffing and character stuffing						
2	Course Outcomes	Simulate the networking concepts and protocols using C/C++.						
3	Aim	Write a program for HDLC frame to perfom Bit Stuffing						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	Bit stuffing is the process of inserting noninformation bits into data to break up bit patterns to affect the synchronous transmission of information. It is widely used in network and communication protocols, in which bit stuffing is a required part of the transmission process. Bit stuffing is commonly used to bring bit streams up to a common transmission rate or to fill frames. Bit stuffing is also used for run-length limited coding. Character stuffing Since this can interfere with the framing, a technique calledcharacter stuffing is used. The sender's data link layer inserts an ASCII DLEcharacter just before the DLE character in the data. The receiver's data link layer removes this DLE before this data is given to the network layer.						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	1) Turn on the computer and launch the application, 2) Open gedit editor and type program. Program name should have the proper extension . 3) Save the program and close the file. 4)Run the simulation program. 5)Now press the play button in the simulation window and the simulation will begins. 6)After simulation is completed see the output in the output window, 7) Write down the output wave forms. 8)Close the file and then start with new program computation. **Algorithm**: 1. read messsage a[]: i=ponter a[] flag: “01111110” j=pointer b[]						

count =0

2. Add flag to b[]

3. Read the bit of a[ ]

4. Save it in b [ ]

5. if (bit = 1) count ++

if (count =6) add ‘0’ to b[ ] , make count = 0

if (bit =0) count = 0

6. Repeat step 3,4 and 5 for all bits of a[ ].

Program:

BIT stuffing:

```
#include<string.h>
#include<stdio.h>

main()
 {
   char a[20],fs[50]="",t[6],r[5];
   int i,j,p=0,q=0;

   printf("enter bit string : ");
   scanf("%s",a);
   strcat(fs,"01111110");

   if(strlen(a)<6)
     {
      strcat(fs,a);
     }
   else
     {
      for(i=0;i<strlen(a)-5;i++)
             {
               for(j=i;j<i+6;j++)
                 {
                   t[p++]=a[j];
                 }
               t[p]='\0';
               if(strcmp(t,"011111")==0)
                 {
                   strcat(fs,"0111110");
                   i=j-1;
                 }
               else
                 {
                   r[0]=a[i];
                   r[1]='\0';
                   strcat(fs,r);
```

```
              }
             p=0;
           }
   for(q=i;q<strlen(a);q++)
     {
           t[p++]=a[q];
     }
   t[p]='\0';
   strcat(fs,t);
  }
  strcat(fs,"01111110");
  printf("After stuffing : %s",fs);

  getch();
}
```

CHARACTER stuffing:

```
#include<string.h>
#include<stdio.h>

main()
{
  char a[30],fs[50]="",t[3],sd,ed,x[3],s[3],d[3],y[3];
  int i,j,p=0,q=0;

  printf("Enter characters to be stuffed : ");
  scanf("%s",a);
  printf("\nEnter a character that represents starting delimiter : ");
  scanf(" %c",&sd);
  printf("\nEnter a character that represents ending delimiter   : ");
  scanf(" %c",&ed);
  x[0]=s[0]=s[1]=sd;
  x[1]=s[2]='\0';
  y[0]=d[0]=d[1]=ed;
  d[2]=y[1]='\0';
  strcat(fs,x);

  for(i=0;i<strlen(a);i++)
   {
     t[0]=a[i];
     t[1]='\0';
     if(t[0]==sd)
          strcat(fs,s);
     else
          if(t[0]==ed)
            strcat(fs,d);
    else
          strcat(fs,t);
```

		}
		 strcat(fs,y);
		 printf("\nAfter stuffing : %s",fs);
		
		 getch();
		 }

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
0	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Plesiochronous digital hierarchy, Synchronous digital hierarchy
13	Remarks	
14	Faculty Signature with Date	

Experiment 08 : Distance vector algorithm to find suitable path for transmission

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Distance vector algorithm to find suitable path for transmission						
2	Course Outcomes	Simulate the networking concepts and protocols using C/C++.						
3	Aim	Write an program for Distance vector algorithm to find suitable path for transmission						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	A distance-vector routingprotocol in data networks determines the best route for data packets based on distance. Distance-vector routing protocols measure the distance by the number of routers a packet has to pass, one router counts as one hop.						
6	Procedure, Program, Activity, Algorithm, Pseudo Code	1) Turn on the computer and launch the application, 2) Open gedit editor and type program. Program name should have the proper extension . 3) Save the program and close the file. 4)Run the simulation program. 5)Now press the play button in the simulation window and the simulation will						

begins.

6)After simulation is completed see the output in the output window,

7) Write down the output wave forms.

8)Close the file and then start with new program computation.

Alogorithm:

1. read the no. Of nodes from graph

2. Maintain routing table for all the nodes

3. initialise distance between the same node= 0 and previous node as the same node.

4. update the routing table.

Program:

```
#include<stdio.h>
#include<stdlib.h>
#define nul 1000
#define nodes 10
int no;
struct node
{
int a[nodes][3];
}
router[nodes];
void init(int r)
{
int i;
for(i=1;i<=no;i++)
{
router[r].a[i][1]=i;
router[r].a[i][2]=999;
router[r].a[i][3]=nul;
}
router[r].a[r][2]=0;
router[r].a[r][3]=r;
}
void inp(int r)
{
int i;
printf("\n enter dist to the node %d to other nodes",r);
printf("\n please enter 999 if there is no direct route\n");
for(i=1;i<=no;i++)
{
if(i!=r)
{
printf("\n enter dist to the node %d:",i);
scanf("%d",&router[r].a[i][2]);
router[r].a[i][3]=i;
```

```
}
}
}
void display(int r)
{
int i;
printf("\n\n the routimg table for node %d is as follows",r);
for(i=1;i<=no;i++)
{
if(router[r].a[i][2]>=999)
printf("\n\t\t\t %d \t no link \t no hop",router[r].a[i][1]);
else
printf("\n\t\t\t %d \t %d \t\t %d",router[r].a[i][1],router[r].a[i][2],router[r].a[i][3]);
}
}
void dv_algo(int r)
{
int i,j,z;
for(i=1;i<=no;i++)
{
if(router[r].a[i][2]!=999 && router[r].a[i][2]!=0)
{
for(j=1;j<=no;j++)
{
z=router[r].a[i][2]+router[i].a[j][2];
if(router[r].a[j][2]>z)
{
router[r].a[j][2]=z;
router[r].a[j][3]=i;
}
}
}
}
}
void find(int x,int y)
{
if(router[x].a[y][3]!=y)
{
find(x,router[x].a[y][3]);
printf("%d-->",router[x].a[y][3]);
find(router[x].a[y][3],y);
return;
}
}
int main()
{
int i,j,x,y;
int choice;
printf("enter the no of nodes required(less than 10 pls):");
scanf("%d",&no);
for(i=1;i<=no;i++)
```

		{
		init(i);
		inp(i);
		}
		printf("\n the configuration of the nodes after initilization is as follows:");
		for(i=i;i<=no;i++)
		display(i);
		for(j=1;j<=no;j++)
		for(i=1;i<=no;i++)
		dv_algo(i);
		printf("\n the configuration of the nodes after computation of path is as follows:");
		for(i=1;i<=no;i++)
		display(i);
		while(1)
		{
		printf("\n\n enter 1 to continue 0 to quit:");
		scanf("\%d",&choice);
		if(choice!=1)
		break;
		printf("\n enter the nodes btn which shortest path is to be found:\n");
		scanf("%d%d",&x,&y);
		printf("\n the shortest path is:");
		printf("%d-->",x);
		find(x,y);
		printf("%d",y);
		printf("\n the length of the shortest path is%d",router[x].a[y][2]);
		}
		return 0;
		}

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Protocol network, LAN, ARPANET routing
13	Remarks	
14	Faculty Signature with Date	

Experiment 09 : Dijkstra's Algorithm to compute shortest path

-	Experiment No.:	1	Marks		Date Planned		Date Conducted	
1	Title	Dijkstra's Algorithm to compute shortest path						
2	Course Outcomes	Simulate the networking concepts and protocols using C/C++.						
3	Aim	Implement Dijkstra's Algorithm to compute shortest routing path.						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	1) Turn on the computer and launch the application, 2) Open gedit editor and type program. Program name should have the proper extension . 3) Save the program and close the file. 4)Run the simulation program. 5)Now press the play button in the simulation window and the simulation will begins. 6)After simulation is completed see the output in the output window, 7) Write down the output wave forms. 8)Close the file and then start with new program computation. **Algorithm:** 1. read no. Of nodes in graph 2. choose start node let current node = start node 3. mark the current node as visited 4. set path length of non visited neighbours of current nodes 5. update the path length of non visited neighbours of current nodes 6. select the node with minimum path length 7. set current nodes = node with minimum path length and mark their current node as visited 8. Repeat the step 5 & 7, (n-1) times.						
		Program: #include<stdio.h>						

```
void sort(void);

static int dsp[10][10],nodes;
struct{
  char src;
  char dest;
  int length;
}stemp,permanent[10]={' ',' ',0},temp[10]={' ',' ',-1};
static int perm,tem;

void main()
{
  int i,j,k,l,m,n=0,point;
  char initial,dest,path[10]={' '};

  printf("\t\t Shortest Path (Dijkstra's algorithm)");
  printf("\n*******************************************************");
  printf("\nEnter the number of nodes:");
  scanf("%d",&nodes);
  printf("\nEnter the adjacency matrix for the graph:\n");

  for(i=0;i<nodes;i++)
  {
       for(j=0;j<nodes;j++)
                         scanf("%d",&dsp[i][j]);
  }
  fflush(stdin);
  printf("\n enter the source node:");
  scanf("%c",&initial);fflush(stdin);
  printf("\n Enter the destination node:");
  scanf("%c",&dest);
  permanent[perm].src=initial;
  permanent[perm].dest=initial;
```

```
permanent[perm++].length=0;
i=permanent[perm-1].dest-97;

for(j=0;j<nodes;j++)
{
    if(i!=j)
    {
      if(dsp[i][j]>0)
      {
           temp[tem].src=permanent[perm-1].src;
           temp[tem].dest=j+97;
           temp[tem++].length=dsp[i][j];
      }
    }
}
sort();

while(tem>=0)
{
j=permanent[perm-1].dest-97;
    for(i=0;i<nodes;i++)
    {
        if(i!=initial-97)
        {
            if(dsp[j][i]>0)
            {
l=-1;
                for(k=0;k<perm;k++)
               {
                if(permanent[k].dest==(i+97))
l=k;
            }
                for(k=0;k<=tem;k++)
                {
```

```
if(temp[k].dest==(i+97))
l=k;
}
if(l<0)
{
temp[tem].src=j+97;
temp[tem].dest=i+97;
for(m=0;m<perm;m++)
{
if(permanent[m].dest==temp[tem].src)
n=permanent[m].length;
}
temp[tem++].length=dsp[j][i]+n;
}

else
{
for(m=0;m<perm;m++)
{
if(permanent[m].dest==j+97)
{
n=permanent[m].length+dsp[j][i];break;
}
else
n=dsp[j][i];
}
if((n<temp[l].length))
{
temp[l].length=n;
temp[l].src=j+97;
temp[l].dest=i+97;
}
}
}
```

```
      }
    }
    sort();
 }
  printf("\nShortest path:\n");
  printf("From %c to %c is:",initial,dest);

  for(i=0;i<perm-1;i++)
  {
      if(permanent[i].dest==dest)
      {
            point=i;n=i;      break;
      }
  }   i=0;

  for(j=perm;j>0;j--)
  {
       if(permanent[j-1].dest==permanent[point].src)
               {
path[i++]=permanent[point].dest;
point=j-1;
  }
  }
  path[i]=initial;
  for(j=i;j>=0;j--)
  printf("%c ",path[j]);
  printf("\t length=%d",permanent[n].length);
  getch();
 }

void sort()
{
int i,j,k;
          for(i=0;i<=tem;i++)
```

		{
		k=1;
		for(j=0;j<=tem;j++)
		{
		if((temp[j].length <= temp[j+1].length))
		{
		stemp=temp[j];
		temp[j]=temp[j+1];
		temp[j+1]=stemp; k=0;
		}
		}
		if(k)
		break;
		}
		
		}

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	vertices in a weighted graph, Google maps, Routing systems
13	Remarks	
14	Faculty Signature with Date	

Experiment 10 : Using CRC-CCITT polynomial to obtain CRC code

-	**Experiment No.:**	1	**Marks**		**Date**		**Date**

					Planned		**Conducted**	

1	Title	Using CRC-CCITT polynomial to obtain CRC code
2	Course Outcomes	Model the networks for different configuration and analyze the results.
3	Aim	For the given data use CRC-CCITT polynomial to obtain CRC code. Verify the program for the cases a) without error b) with error
4	Material / Equipment Required	Lab Manual
5	Theory, Formula, Principle, Concept	Whenever digital data is stored or interfaced, data corruption might occur. Since the beginning of computer science, developers have been thinking of ways to deal with this type of problem. For serial data they came up with the solution to attach a parity bit to each sent byte. This simple detection mechanism works if an odd number of bits in a byte changes, but an even number of false bits in one byte will not be detected by the parity check. To overcome this problem developers have searched for mathematical sound mechanisms to detect multiple false bits. The CRC calculation or cyclic redundancy check was the result of this. Nowadays CRC calculations are used in all types of communications. All packets sent over a network connection are checked with a CRC. Also each data block on your hard disk has a CRC value attached to it. Modern computer world cannot do without these CRC calculations. So let's see why they are so widely used. The answer is simple; they are powerful, detect many types of errors and are extremely fast to calculate especially when dedicated hardware chips are used.

```
                  1 0 1 = 5
            -------------
1 0 0 1 1 / 1 1 0 1 1 0 1
            1 0 0 1 1 | |
            --------- | |
              1 0 0 0 0 |
              0 0 0 0 0 |
              --------- |
              1 0 0 0 0 1
                1 0 0 1 1
                ---------
                  1 1 1 0 = 14 = remainder
```

EC

Prepared by | Checked by | Approved

6 Procedure, Program, Activity, Algorithm, Pseudo Code

Algorithm:

1. Multiply M(x) by highest power in G(x). i.e. Add So much zeros to M(x).

2. Divide the result by G(x). The remainder = C(x). Special case: This won't work if bitstring =all zeros. We don't allow such an M(x).But M(x) bitstring = 1 will work, for example. Can divide 1101 into 1000.

3. If: x div y gives remainder c that means: x = n y + c Hence (x-c) = n y (x-c) div y gives remainder 0 Here (x-c) = (x+c) Hence (x+c) div y gives remainder 0

4. Transmit: T(x) = M(x) + C(x)

5. Receiver end: Receive T(x). Divide by G(x), should have remainder 0.

Program:

```
#include<stdio.h>
int a[100],b[100],i,j,len,k,count=0;

//Generator Polynomial:g(x)=x^16+x^12+x^5+1
int gp[]={1,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,1,};

int main()
{
        void div();
        printf("\nEnter the length of Data Frame :");
        scanf("%d",&len);
        printf("\nEnter the Message :");
        for(i=0;i<len;i++)
                scanf("%d",&a[i]);

        //Append r(16) degree Zeros to Msg bits
        for(i=0;i<16;i++)
                a[len++]=0;
//Xr.M(x)  (ie. Msg+16 Zeros)
        for(i=0;i<len;i++)
                b[i]=a[i];

        //No of times to be divided ie. Msg Length
        k=len-16;
        div();
        for(i=0;i<len;i++)
                b[i]=b[i]^a[i];       //MOD 2 Substraction
        printf("\nData to be transmitted : ");
        for(i=0;i<len;i++)
                printf("%2d",b[i]);

        printf("\n\nEnter the Reveived Data : ");
        for(i=0;i<len;i++)
                scanf("%d",&a[i]);

        div();
```

```
        for(i=0;i<len;i++)
                if(a[i]!=0)
                {
                        printf("\nERROR in Recived Data");
                        return 0;
                }
        printf("\nData Recived is ERROR FREE");
}

void div()
{
        for(i=0;i<k;i++)
        {
                if(a[i]==gp[0])
                {
                        for(j=i;j<17+i;j++)
                                a[j]=a[j]^gp[count++];
                }
                count=0;
        }
}
```

7 Block, Circuit, Model Diagram

8

	CRC-CCITT	CRC-16	CRC-32
Checksum Width	16 bits	16 bits	32 bits
Generator Polynomial	10001000000100001	11000000000000101	100000100110000010001110110110111

Error detection with CRC

9 Sample Calculations

10	Graphs, Outputs	Enter the length of Data Frame: 4 Enter the Message: 1 0 1 1 Data to be transmitted: 1 0 1 1 1 0 1 1 0 0 0 1 0 1 1 0 1 0 1 1 Enter the Reveived Data: 1 0 1 1 1 0 1 1 0 0 0 0 0 1 1 0 1 0 1 1 ERROR in Recived Data
11	Results & Analysis	
12	Application Areas	Digital networks and storage devices
13	Remarks	
14	Faculty Signature with Date	

Experiment 11 : Implementation of Stop & Wait protocol

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Implementation of Stop & Wait protocol						
2	Course Outcomes	Model the networks for different configuration and analyze the results.						
3	Aim	Implementation of Stop & Wait protocol and sliding window protocol						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept							
6	Procedure, Program, Activity, Algorithm, Pseudo Code	1) Turn on the computer and launch the application, 2) Open g-edit editor and type program. Program name should have the proper extension . 3) Save the program and close the file. 4)Run the simulation program. 5)Now press the play button in the simulation window and the simulation will begins. 6)After simulation is completed see the output in the output window, 7) Write down the output wave forms. 8)Close the file and then start with new program computation. **Algorithm**: 1. Read number of frame 2. send a frame 3. set the timmer 4. if acknowledgement recieved before timeout, send the next frame 5. Else send the frame again 6. repeat 2,3,4,5 for all the frames.						

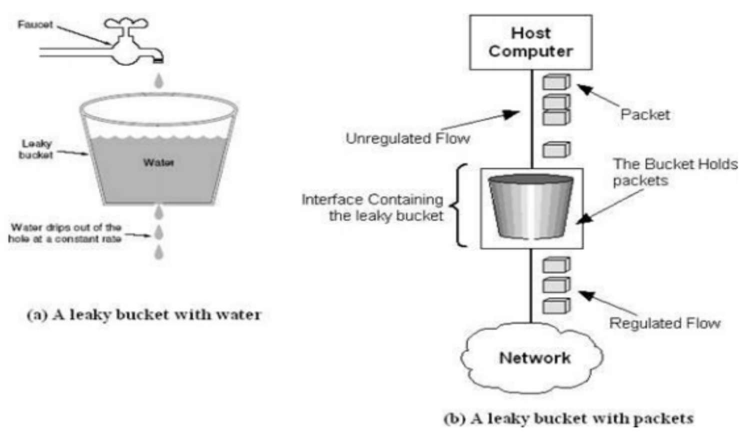
		Program:
		
		`#include<stdio.h>`
		`#include<conio.h>`
		`#include<stdlib.h>`
		`void main()`
		`{`
		`    int i,j,noframes,x,x1=10,x2;`
		`    noframes=10;`
		`    i=1;`
		`    j=1;`
		`        printf("number of frames is %d ",noframes);`
		
		`    getch();`
		`    while(noframes>0)`
		`    {`
		`        printf("\nsending frames is %d",i);`
		`        x=rand()%10;`
		`        if(x%10==0)`
		`        {`
		`            for(x2=1;x2<2;x2++)`
		`            {`
		`                printf("\n waiting for %d seconds\n",x2);`
		`                sleep(x2);`
		`            }`
		`            printf("\n sending frames %d\n",i);`
		`            x=rand()%10;`
		`        }`
		`        printf("\n ack for frame %d\n",j);`
		`        noframes=noframes-1;`
		`i++;`
		`j++;`
		`    }`
		`    printf("\n end of stop and wait protocol\n");`
		
		`}`

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	

10	Graphs, Outputs	
11	Results & Analysis	
12	Application Areas	Telecommunications, IBM Binary Synchronous Communication
13	Remarks	
14	Faculty Signature with Date	

Experiment 12 : Congesion control using Leakg bucket algorithm

-	**Experiment No.:**	1	**Marks**		**Date Planned**		**Date Conducted**	
1	Title	Congesion control using Leakg bucket algorithm						
2	Course Outcomes	Model the networks for different configuration and analyze the results.						
3	Aim	Write a program for congesion control using leaky bucket algorithm						
4	Material / Equipment Required	Lab Manual						
5	Theory, Formula, Principle, Concept	The main concept of the leaky bucket algorithm is that the output data flow remains constant despite the variant input traffic, such as the water flow in a bucket with a small hole at the bottom. In case the bucket contains water (or packets) then the output flow follows a constant rate, while if the bucket is full any additional load will be lost because of spillover. In a similar way if the bucket is empty the output will be zero. From network perspective, leaky bucket consists of a finite queue (bucket) where all the incoming packets are stored in case there is space in the queue, otherwise the packets are discarded. In order to regulate the output flow, leaky bucket transmits one packet from the queue in a fixed time (e.g. at every clock tick). In the following figure we can notice the main rationale of leaky bucket algorithm, for both the two approaches (e.g. leaky bucket with water (a) and with packets (b)).							



(a) A leaky bucket with water

(b) A leaky bucket with packets

6	Procedure, Program, Activity, Algorithm, Pseudo Code	1) Turn on the computer and launch the application, 2) Open g-edit editor and type program. Program name should have the proper extension . 3) Save the program and close the file. 4)Run the simulation program. 5)Now press the play button in the simulation window and the simulation will begins. 6)After simulation is completed see the output in the output window, 7) Write down the output wave forms. 8)Close the file and then start with new program computation. **Algorithm**: Steps: 1. Read The Data For Packets 2. Read The Queue Size 3. Divide the Data into Packets 4. Assign the random Propagation delays for each packets to input into the bucket (input_packet). 5. wlile((Clock++<5*total_packets)and (out_packets< total_paclets)) a. if (clock == input_packet) i. insert into Queue b. if (clock % 5 == 0) i. Remove paclet from Queue 6. End

Program:

```
#include<stdio.h>
#include<stdlib.h>
#define MIN(x,y) (x>y)?y:x
int main()
{
int orate,drop=0,cap,x,count=0,inp[10]={0},i=0,nsec,ch;
printf("\n enter bucket size : ");
scanf("%d",&cap);
printf("\n enter output rate :");
scanf("%d",&orate);
do
{
printf("\n enter number of packets coming at second %d :",i+1);
scanf("%d",&inp[i]);
i++;
printf("\n enter 1 to contiue or 0 to quit.........");
scanf("%d",&ch);
}
```

```
while(ch);
nsec=i;
printf("\n second \t recieved \t sent \t dropped \t remained \n");
for(i=0;count || i<nsec;i++)
 {
printf("   %d",i+1);
printf(" \t%d\t ",inp[i]);
printf(" \t %d\t ",MIN((inp[i]+count),orate));
 if((x=inp[i]+count-orate)>0)
  {
if(x>cap)
{
count=cap;
drop=x-cap;
}
else
{
count=x;
drop=0;
}
}
else
{
drop=0;
count=0;
}
printf(" \t %d                    \t %d \n",drop,count);
}
return 0;
}
```

7	Block, Circuit, Model Diagram, Reaction Equation, Expected Graph	
8	Observation Table, Look-up Table, Output	
9	Sample Calculations	
10	Graphs, Outputs	Output: Enter The Bucket Size 5 Enter The Operation Rate 2 Enter The No. Of Seconds You Want To Stimulate 3 Enter The Size Of The Packet Entering At 1 sec 5

		Enter The Size Of The Packet Entering At 1 sec 4 Enter The Size Of The Packet Entering At 1 sec 3 Second|Packet Recieved|Packet Sent|Packet Left|Packet Dropped| -- 1 5 2 3 0 2 4 2 3 2 3 3 2 3 1 4 0 2 1 0 5 0 1 0 0
11	Results & Analysis	
12	Application Areas	The leaky bucket as a meter can be used in either traffic shaping or traffic policing, The leaky bucket algorithm as a meter can also be used in a leaky bucket counter to measure the rate of random or stochastic processes.
13	Remarks	
14	Faculty Signature with Date	

VIVA QUESTIONS

1. What are functions of different layers?
2. Differentiate between TCP/IP Layers and OSI Layers
3. Why header is required?
4. What is the use of adding header and trailer to frames?
5. What is encapsulation
6. Why fragmentation requires?
7. What is MTU?
8. Which layer imposes MTU
9. Differentiate between flow control and congestion control.
10. Differentiate between Point-to-Point Connection and End-to-End connections.
11. What are protocols running in different? layers
12. What is Protocol Stack?
13. Differentiate between TCP and UDP.
14. Differentiate between Connectionless and connection oriented connection.
15. Why frame sorting is required?
16. What is meant by subnet?
17. What is meant by Gateway?
18. What is an IP address?
19. What is MAC address?
20. Why IP address is required when we have MAC address?
21. What is meant by port?

22. What are ephemerical port number and well known port numbers?
23. What is a socket?
24. What are the parameters of socket()?
25. Describe bind(), listen(), accept(),connect(), send() and recv().
26. What are system calls? Mention few of them.
27. What is IPC? Name three techniques.
28. Explain mkfifo(), open(), close() with parameters.
29. What is meant by file descriptor?
30. What is meant by traffic shaping?
31. How do you classify congestion control algorithms?
32. Differentiate between Leaky bucket and Token bucket.
33. How do you implement Leaky bucket?
34. How do you generate busty traffic?
35 What is the polynomial used in CRC-CCITT?
36. What are the other error detection algorithms?
37. What is difference between CRC and Hamming code?
38. Why Hamming code is called 7,4 code?
39. What is odd parity and even parity?
40. What is meant by syndrome?
41. What is generator matrix?
42. What is spanning tree?
43. Differentiate between Prim's and Kruskal's algorithm.
44. What are Routing algorithms?
45. How do you classify routing algorithms? Give examples for each.
46. What are drawbacks in distance vector algorithm?
47. How routers update distances to each of its neighbor?
48. How do you overcome count to infinity problem?
49. What is cryptography?
50. How do you classify cryptographic algorithms?
51. What is public key?
52. What is private key?
53. What are key, ciphertext and plaintext?
54. What is simulation?
55. What are advantages of simulation?
56. Differentiate between Simulation and Emulation.
57. What is meant by router?
58. What is meant by bridge?
59. What is meant by switch?
60. What is meant by hub?
61. Differentiate between route, bridge, switch and hub.
62. What is ping and telnet?
63. What is FTP?
64. What is BER?
65. What is meant by congestion window?

66. What is BSS?
67. What is incoming throughput and outgoing throughput?
68. What is collision?
69. How do you generate multiple traffics across different sender-receiver pairs?
70. How do you setup Ethernet LAN?
71. What is meant by mobile host?
72. What is meant by NCTUns?
73. What are dispatcher, coordinator and nctunsclient?
74. Name few other Network simulators
75. Differentiate between logical and physical address.
76. Which address gets affected if a system moves from one place to another place?
77. What is ICMP? What are uses of ICMP? Name few.
78. Which layer implements security for data?