

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB RECORD

Computer Network Lab (23CS5PCCON)

Submitted by

Ankit Singh Bhatti (1BM22CS353)

in partial fulfillment for the award of the degree of

**BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING**



B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

Academic Year 2024-25 (odd)

B.M.S. College of Engineering

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the Lab work entitled “Computer Network (23CS5PCCON)” carried out by **Ankit Singh Bhatti (1BM22CS353)**, who is bonafide student of **B.M.S. College of Engineering**. It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements of the above-mentioned subject and the work prescribed for the said degree.

Prof. Sneha N P Assistant Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor & HOD Department of CSE, BMSCE
--	--

Index

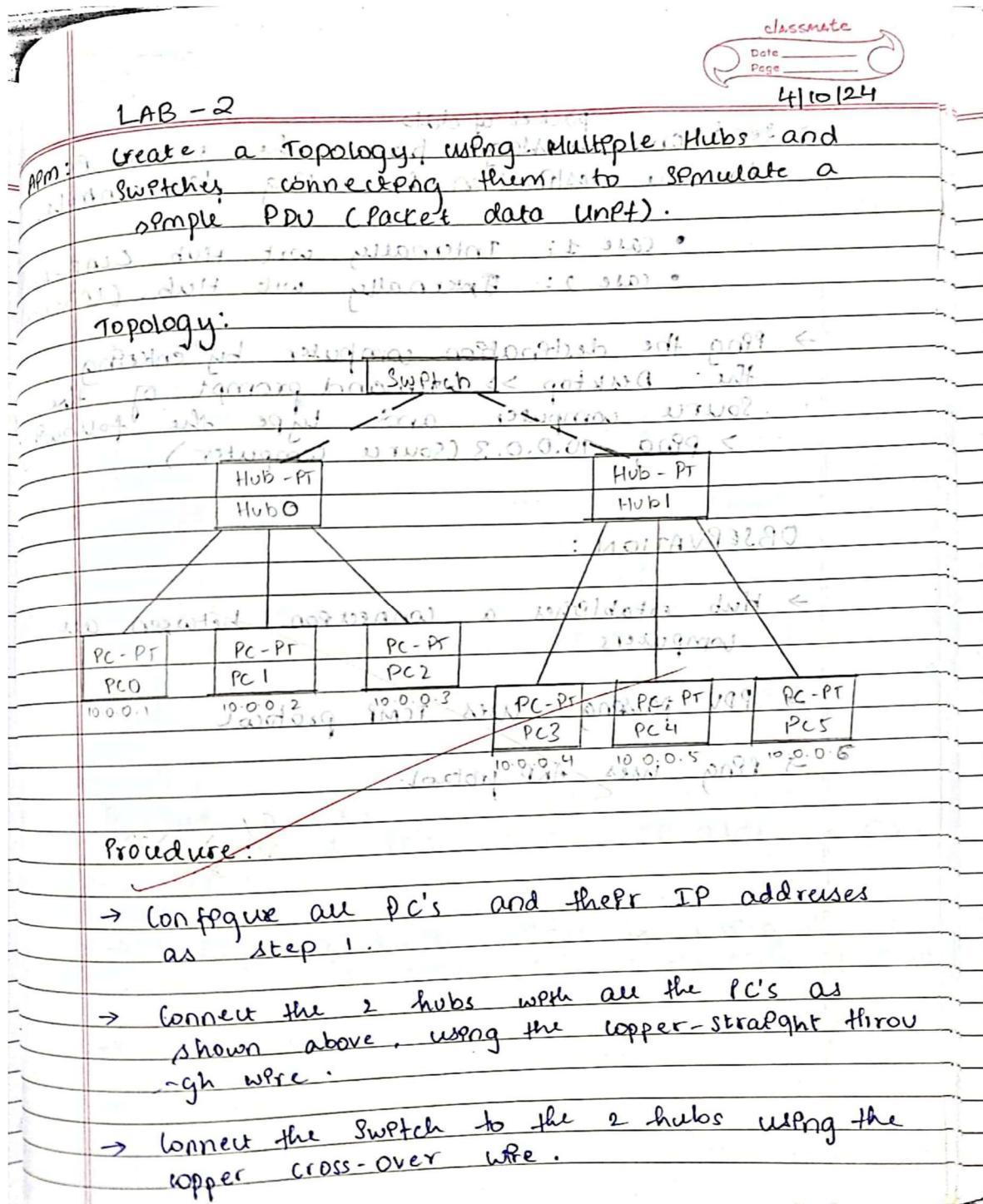
Sl. No.	Date	Experiment Title	Page No.
1	4-10-24	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.	1-4
2	18-10-24	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply	5-9
3	25-10-24	Configure default route, static route to the Router	10-15
4	8-11-24	Configure DHCP within a LAN and outside LAN.	16-19
5	22-11-24	Configure RIP routing Protocol in Routers	20-24
6	27-11-24	Configure OSPF routing protocol	25-27
7	22-11-24	Demonstrate the TTL/ Life of a Packet	28-31
8	8-11-24	Configure Web Server, DNS within a LAN.	32-34
9	20-12-24	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)	35-38
10	20-12-24	To understand the operation of TELNET by accessing the router in server room from a PC in IT office.	39-42
11	20-12-24	To construct a VLAN and make the PC's communicate among a VLAN	43-47
12	20-12-24	To construct a WLAN and make the nodes communicate wirelessly	48-52
		CYCLE - 2	
1	15-11-24	Write a program for error detecting code using CRC-CCITT (16-bits).	53-55
2	15-11-24	Write a program for congestion control using Leaky bucket algorithm.	56-58
3	20-12-24	Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	59-61

4	20-12-24	Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	62-64
---	----------	--	-------

Github Link: <https://github.com/AnkitSB19/CNSLAB>

Program 1

- i. Aim of the program: Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.
- ii. Procedure along with the topology:



Q. 10)

→ Send one ~~message~~ ^{packet of data} from the source PC (LAB-2-1) to the destination PC (LAB-2-2) using play controls.

- Case 1: Internally wrt Hub (LAB-2-1)
- Case 2: Externally wrt Hub. (LAB-2-2)

→ Ping the destination computer by entering the Desktop >> Command prompt of the Source computer and type the following:
> ping 10.0.0.3 (source computer).

OBSERVATION :

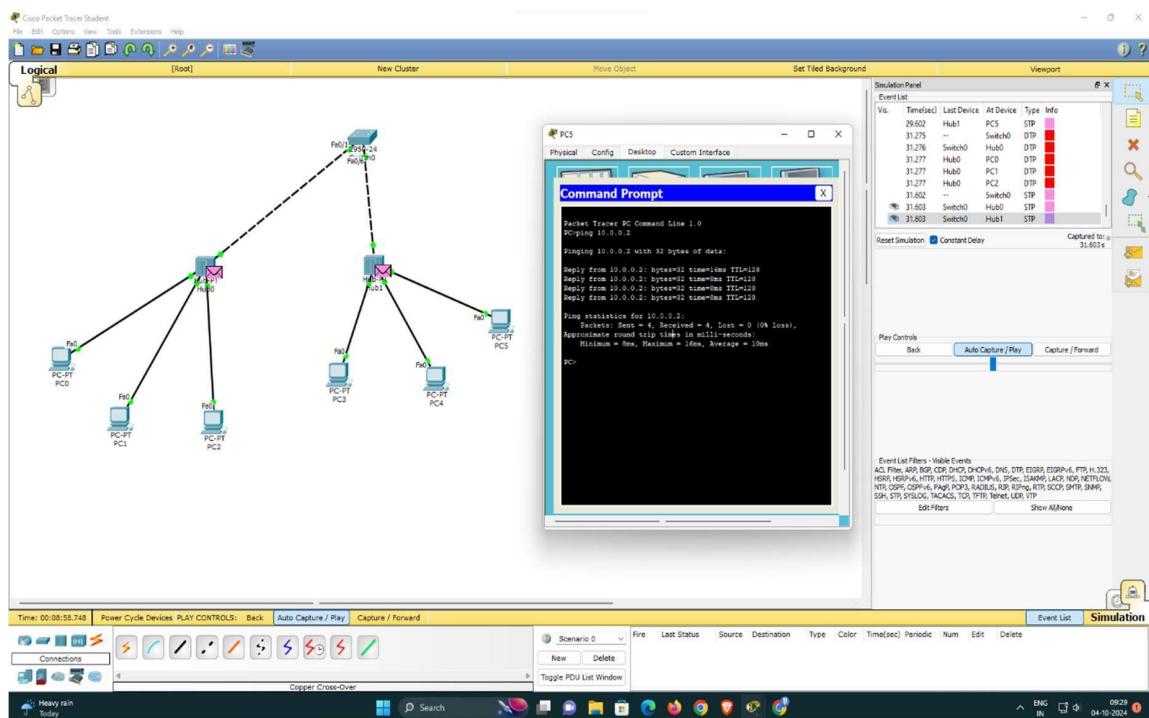
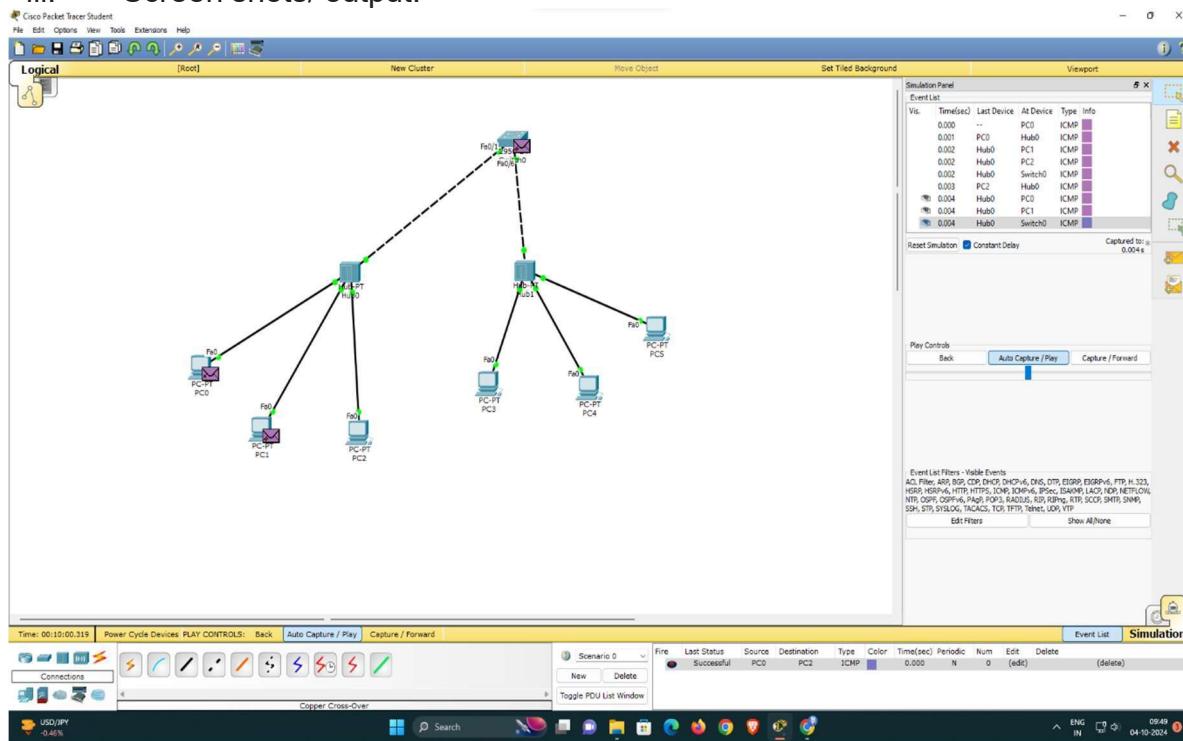
→ Hub establishes a connection between all computers

→ PDV passing uses ICMP protocol

→ PING uses ARP protocol.

Q. 10)

iii. Screen shots/ output:



iv. Observation:

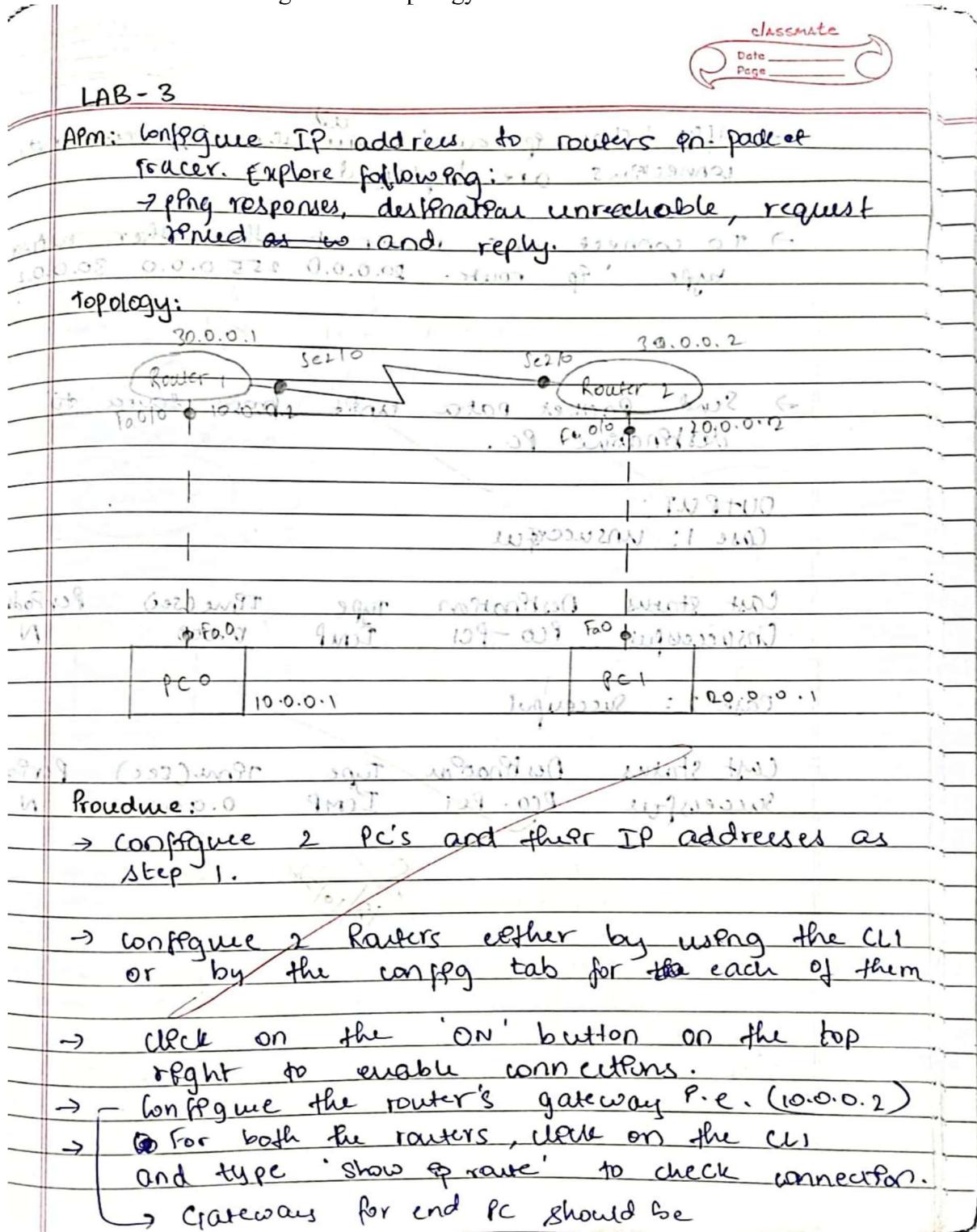
- Send one packet of data from the source PC (LAB-2-1) to the destination PC using play controls.
- case 1: Internally wrt Hub (LAB-2-1)
 - case 2: Externally wrt Hub. (LAB-2-2)
- Ping the destination computer by entering the Desktop > Command prompt of the Source computer and type the following:
→ ping 10.0.0.3 (source computer).

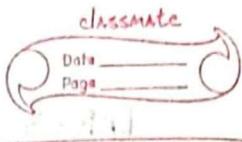
OBSERVATION :

- Hub establishes a connection between all computers
- PDV pumping uses ICMP protocol
- PING uses ARP protocol.

Program 2

- i. Aim of the program: Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply.
- ii. Procedure along with the topology:





→ After 'show ip route' all the different network connections are displayed.

→ To connect one router to other network type 'ip route 20.0.0.0 255.0.0.0 30.0.0.2'

↓ ↓
Network IP subnet mask other networks' IP address

→ Send packet data unit from source to destination pc.

OUTPUT:

Case 1: unsuccessful

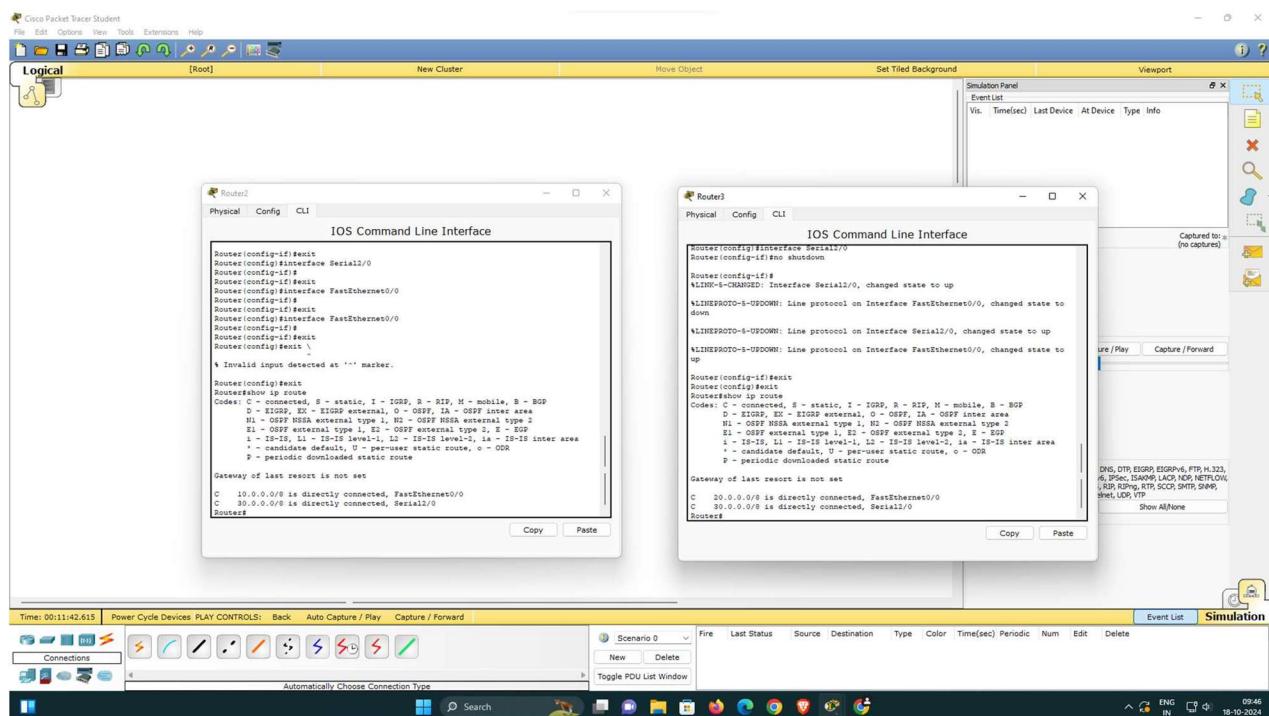
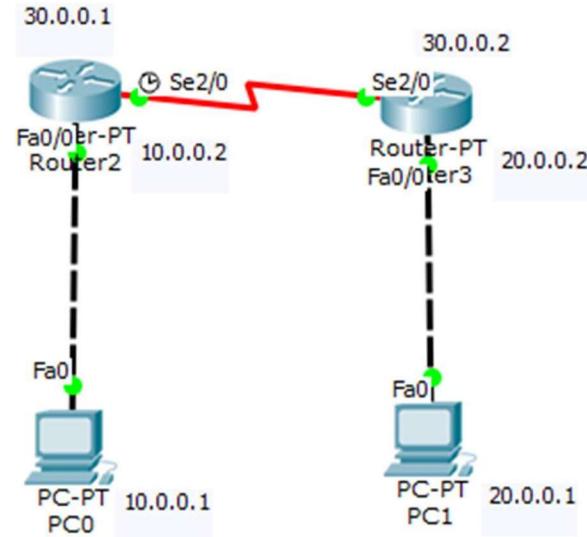
Last Status	Destination	Type	Time(sec)	Per Pdpc
Unsuccessful	PCO - PCI	ICMP	0.000	N

Case 2: successful

Last Status	Destination	Type	Time(sec)	Per Pdpc
Successful	PCO - PCI	ICMP	0.0.0.0.0.0000	N

~~12/10/2022~~
~~12/10/2022~~

iii. Screen shots/ output:



```
Router(config-if)#exit
Router(config)#exit
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

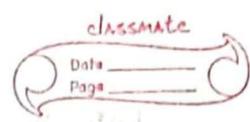
C    20.0.0.0/8 is directly connected, FastEthernet0/0
C    30.0.0.0/8 is directly connected, Serial2/0
Router#cocconfig t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip route 10.0.0.0 255.0.0.0 30.0.0.1
Router(config)#
Router(config)#
SYS-5-CONFIG_I: Configured from console by console

Router(config)#exit
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

S    10.0.0.0/8 [1/0] via 30.0.0.1
C    20.0.0.0/8 is directly connected, FastEthernet0/0
C    30.0.0.0/8 is directly connected, Serial2/0
Router#
SYS-5-CONFIG_I: Configured from console by console
```

iv. Observation:



→ After 'show ip route' all the different network connections are displayed.

→ To connect one router to another network type 'ip route 20.0.0.0 255.0.0.0 30.0.0.2'

↓ ↓
 Network IP subnet mask other networks' IP address

→ Send Packet Data unit from source to destination pc.

OUTPUT:

Case 1: unsuccessful

Last Status	Destination	Type	T Ping (sec)	Per PodPC
Unsuccessful	PC0 - PC1	ICMP	0.000	N

Case 2: Successful

Last Status	Destination	Type	T Ping (sec)	Per PodPC
Successful	PC0 - PC1	ICMP	0.0.0.1 ms	N

→ successful ping between two hosts

✓ 100% .1 ms

→ open and edit route configuration
→ add entry of dest IP and subnet mask

→ no routes found in the ip table
→ add static route entry

(source), add gateway IP value with broadcast address

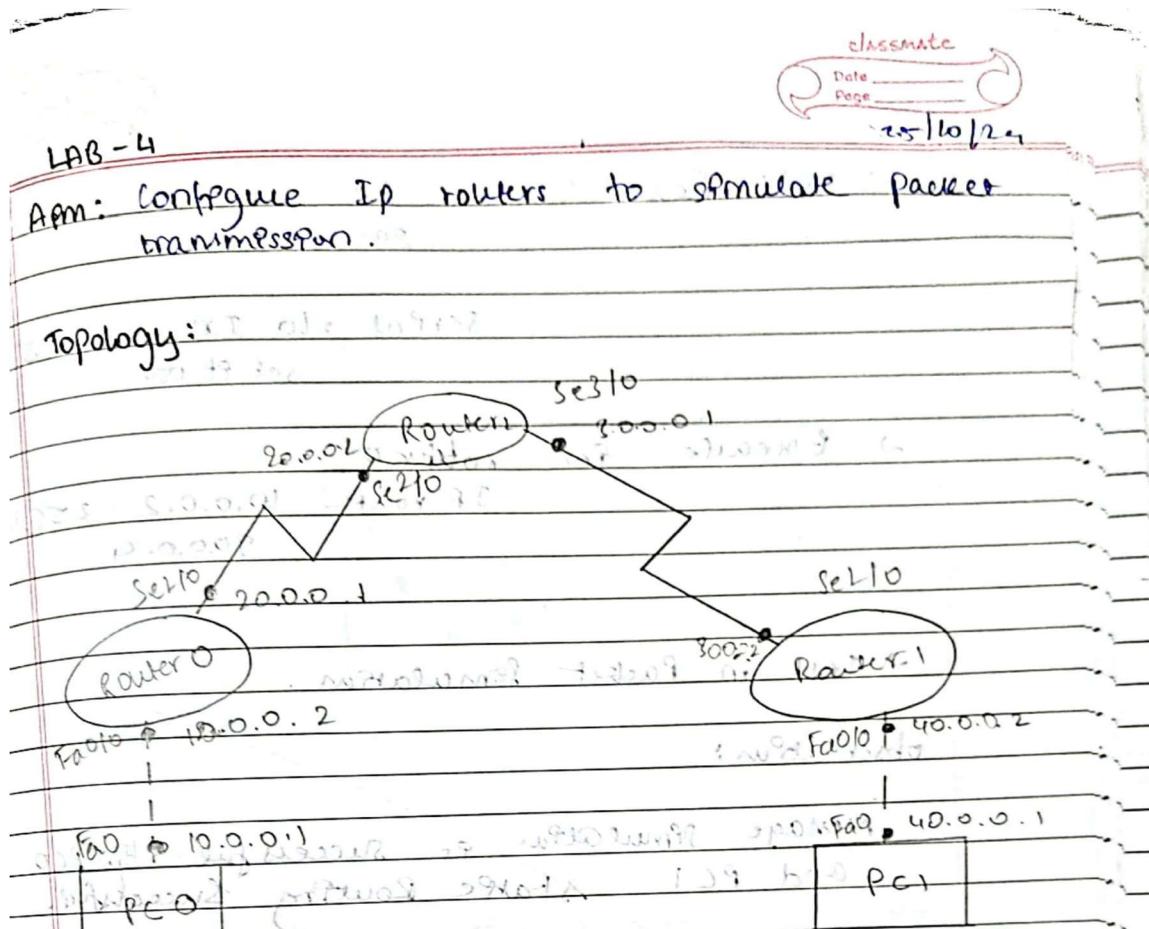
→ add the route, gateway IP must be same as source

→ static route entry added in the ip table

→ check if the route is successful

Program 3:

- Aim of the program: Configure default route, static route to the Router
- Procedure along with the topology



Procedure :

→ Configure PC0 IP = 10.0.0.1
Gateway = 10.0.0.2

→ Configure PC1 IP = 10.0.0.1
Gateway = 20.0.0.2

→ Configure Router 2 Fa0/0 IP = 10.0.0.2
Serial 2/0 = 30.0.0.3
Default ON.

→ Config Router 3 with IP address 10.0.0.2
20.0.0.2

Serial 2/0 IP: = 40.0.0.5
set pt DN

→ Execute in router 0
IP route 10.0.0.2 255.0.0.0
90.0.0.4

→ test in Packet Simulation.

Observation:

Message simulatior is successful w/ PCO
and PLC static Routing successful.

OK

8/11/11

10:16:08

10.0.0.1 → 91.0.1.1 received

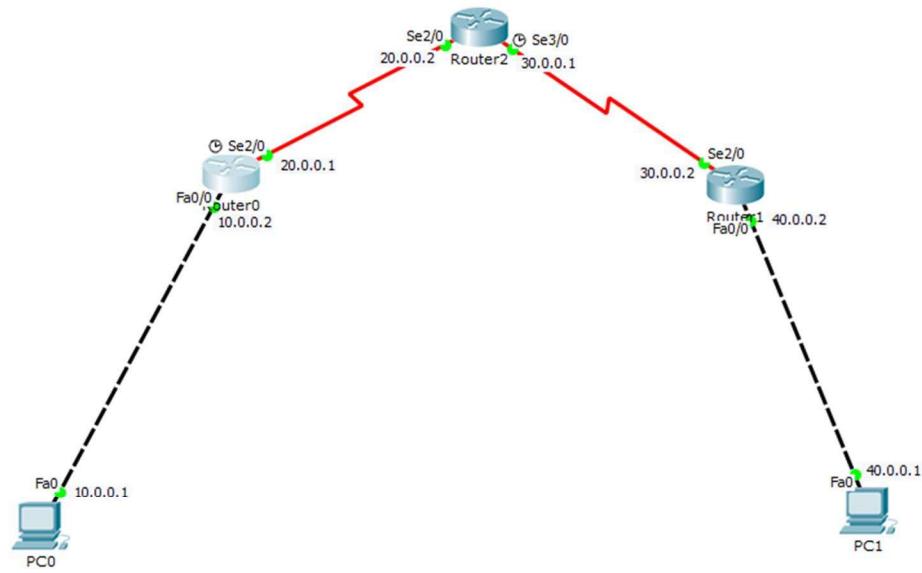
10.0.0.1 → 91.0.1.1

10.0.0.1 → 91.0.1.1 received

10.0.0.1 → 91.0.1.1

10.0.0.1 → 91.0.1.1

iii. Screen shots/ output:



 Router1

Physical Config CLI

```
Router(config-if)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to up

Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 40.0.0.2 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to up

Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#
Router(config-if)#exit
Router(config-if)#
Router(config-if)#exit
Router(config)#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

C    30.0.0.0/8 is directly connected, Serial2/0
C    40.0.0.0/8 is directly connected, FastEthernet0/0
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip route 20.0.0.0 255.0.0.0 30.0.0.1
Router(config)#ip route 10.0.0.0 255.0.0.0 30.0.0.1
Router(config)#exit
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

S    10.0.0.0/8 [1/0] via 30.0.0.1
S    20.0.0.0/8 [1/0] via 30.0.0.1
C    30.0.0.0/8 is directly connected, Serial2/0
C    40.0.0.0/8 is directly connected, FastEthernet0/0
Router#
```

Router2

Physical Config CLI

```
% Invalid input detected at '^' marker.

Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
show ip route
^
% Invalid input detected at '^' marker.

Router#show ip route
Translating "'...domain server (255.255.255.255)
% Invalid input detected
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

C    20.0.0.0/8 is directly connected, Serial2/0
C    30.0.0.0/8 is directly connected, Serial3/0
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#ip route 10.0.0.0 255.0.0.0 20.0.0.1
Router(config)#exit
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

S    10.0.0.0/8 [1/0] via 20.0.0.1
C    20.0.0.0/8 is directly connected, Serial2/0
C    30.0.0.0/8 is directly connected, Serial3/0
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#ip route 40.0.0.0 255.0.0.0 30.0.0.2
Router(config)#exit
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

S    10.0.0.0/8 [1/0] via 20.0.0.1
C    20.0.0.0/8 is directly connected, Serial2/0
C    30.0.0.0/8 is directly connected, Serial3/0
S    40.0.0.0/8 [1/0] via 30.0.0.2
```

iv. Observation:

classmate

Date _____

Page _____

2 - 231

→ Config Router 3 with IP

20.0.0.2 Subnet mask

Serial 2 to IP = 140.0.0.5

Set Pt to

→ Execute in router 0

IP route 10.0.0.2 255.0.0.0
30.0.0.4

→ Test in Packet Simulation.

Observation:

Message Simulation is successful by PCO
and PLC Static Routing Successful.

8/11/20
: 2018

10.0.0.1 → 97 0.0 received

10.0.0.1 → 97.0.0

10.0.0.1 → 97 1.0 received

10.0.0.1 → 97.0.0

10.0.0.1 → 97.0.0.1 1.0.0.0.0 received

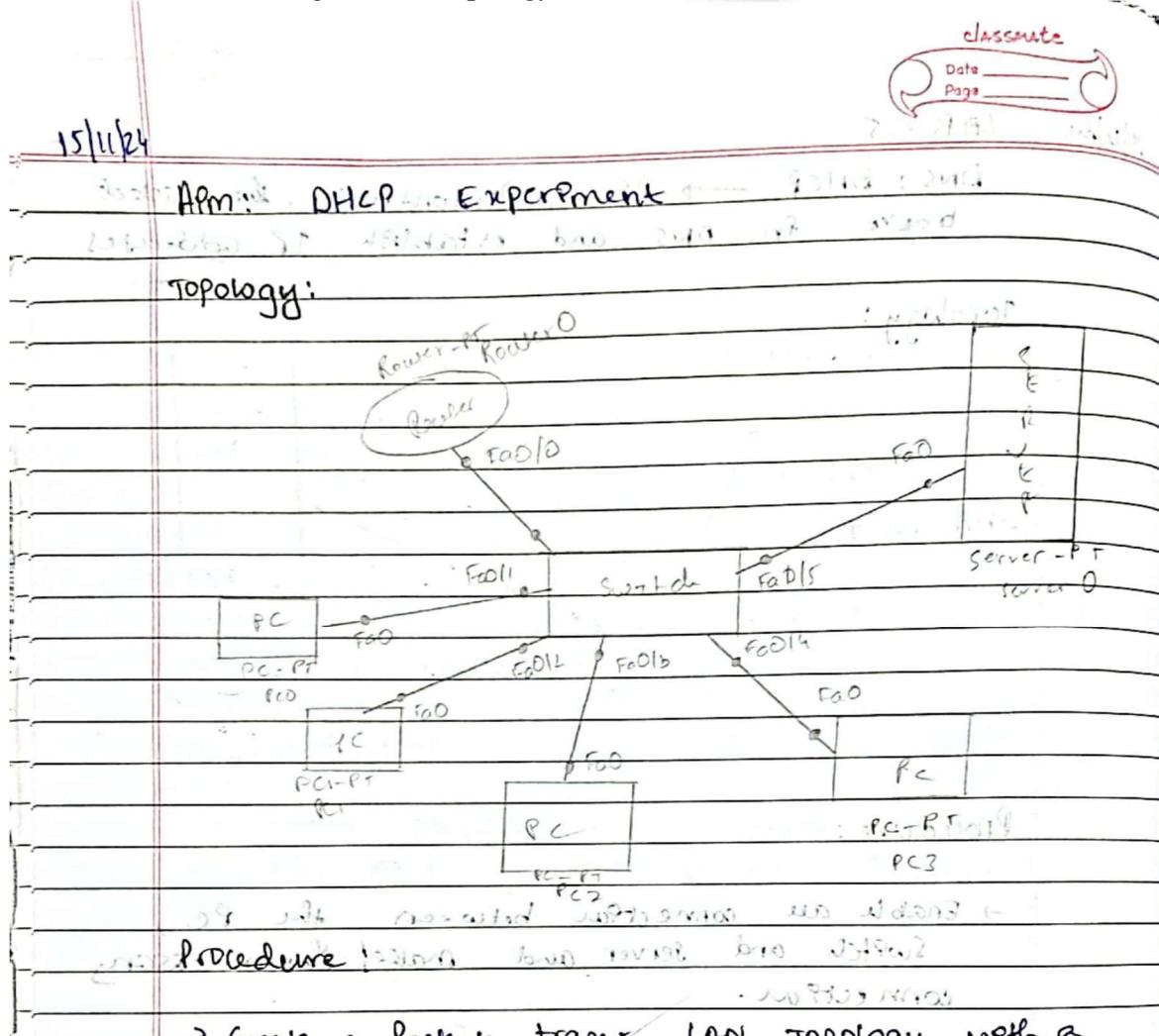
10.0.0.1 → 97.0.0.1

10.0.0.1

Program 4

i. Aim of the program: Configure DHCP within a LAN and outside LAN.

ii. Procedure along with the topology:



Without a DHCP server, pointing to the DHCP server's IP address.

- On client PCs, select "DHCP" in IP configuration to obtain IP addresses automatically.

~~Observation:~~

→ IP addresses are assigned to all the PC's with the help of DHCP.

Forward and return between PC's.

Host task 2021-01-18 17:40:07
was run at 18:00 and ended at 18:00.

Host task 2021-01-18 18:00:00
was run at 18:00 and ended at 18:00.

Host task 2021-01-18 18:00:00
was run at 18:00 and ended at 18:00.

Host task 2021-01-18 18:00:00
was run at 18:00 and ended at 18:00.

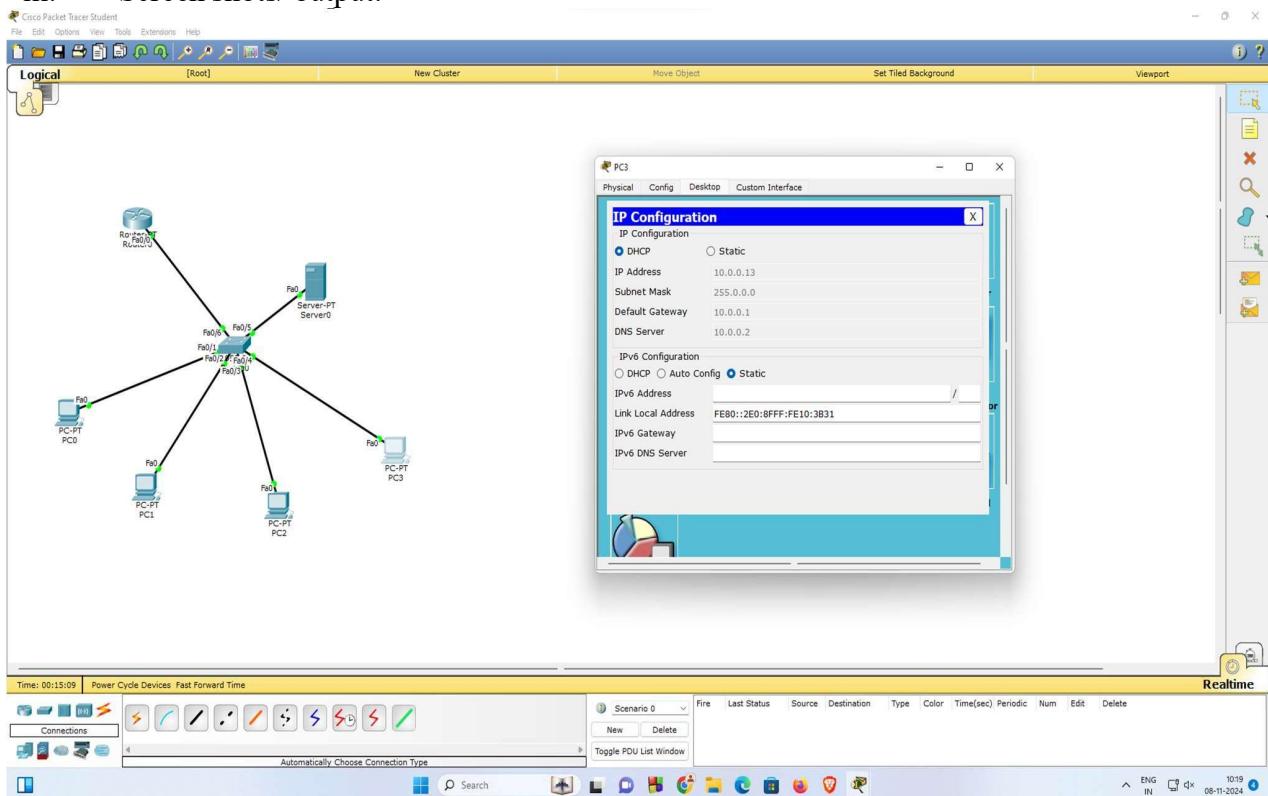
Host task 2021-01-18 18:00:00
was run at 18:00 and ended at 18:00.

Host task 2021-01-18 18:00:00
was run at 18:00 and ended at 18:00.

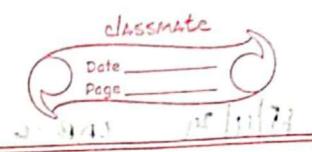
Host task 2021-01-18 18:00:00
was run at 18:00 and ended at 18:00.

Host task 2021-01-18 18:00:00
was run at 18:00 and ended at 18:00.

iii. Screen shots/ output:



iv. Observation:



PC's without a DHCP server, pointing to the DHCP server's IP address.

- On client PCs, select "DHCP" in IP configuration to obtain IP addresses automatically.

~~Observation: set of 2 PCs~~

→ IP addresses are established by all the PCs with the help of DHCP.

Forward has been obtained.

Host 1 IP: 192.168.1.100
Host 2 IP: 192.168.1.101

Forward has been obtained.

Host 1 IP: 192.168.1.100
Host 2 IP: 192.168.1.101

Forward has been obtained.

Host 1 IP: 192.168.1.100
Host 2 IP: 192.168.1.101

Forward has been obtained.

Host 1 IP: 192.168.1.100
Host 2 IP: 192.168.1.101

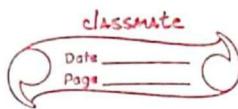
Forward has been obtained.

Program 5

i. Aim of the program: Configure RIP routing Protocol in Routers

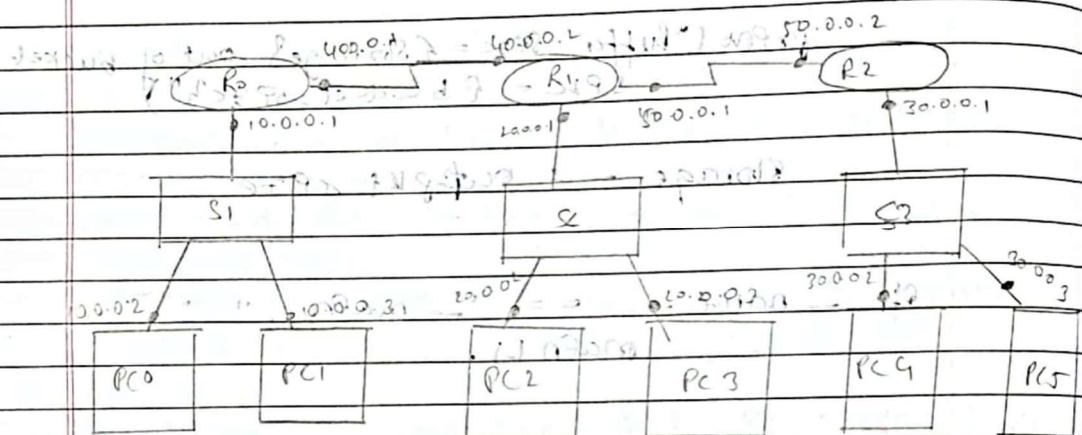
ii. Procedure along with the topology:

22/11/24 LAB - 7



Aim: Demonstrate RIP Routing Information Protocol.

Topology:



Procedure:

1. Configure the two routers.

2. Configure the end devices with respect to IP addresses.

3. Configure routers and their fast Ethernet ports and serial port as shown in the figure.

4. Configure the gateway in end devices.

5. Open the CLI of each router, open config terminal and type 'router rip' after that type all the networks.

nodes a, b, c are connected

→ Simulate the packet transmission from one device to another device.

OBSERVATION:

→ Packet transmission from source to destination is successful.

→ IP addresses are established.

→ TTL originally starts with a value 255

→ TTL decrements at each hop during packet transmission.

→ After 3 hops, TTL becomes 0 & packet is dropped.

→ At 4th hop, TTL becomes -ve & packet is dropped.

→ At 5th hop, TTL becomes 0 & packet is dropped.

→ At 6th hop, TTL becomes -ve & packet is dropped.

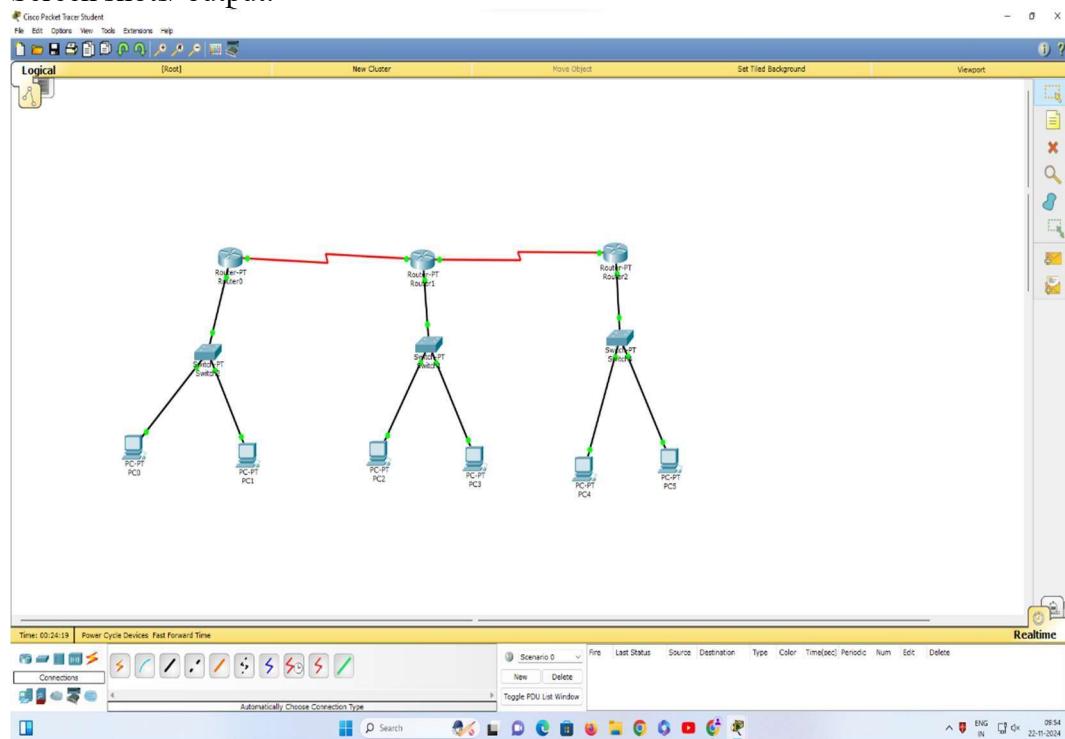
→ At 7th hop, TTL becomes 0 & packet is dropped.

→ At 8th hop, TTL becomes -ve & packet is dropped.

→ At 9th hop, TTL becomes 0 & packet is dropped.

→ At 10th hop, TTL becomes -ve & packet is dropped.

iii. Screen shots/ output:



```

*Mar 21 00:24:19.000 00000000: Line protocol on interface Serial1/0, changed state to up

Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 40.0.0.0
Router(config-router)#network 20.0.0.0
Router(config-router)#network 50.0.0.0
Router(config-router)#exit
Router(config)#exit
Router#
*SYS-5-CONFIG_I: Configured from console by console

Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
      * - candidate default, U - per-user static route, o - ODR
      P - periodic downloaded static route

Gateway of last resort is not set

R    10.0.0.0/8 [120/1] via 40.0.0.1, 00:00:17, Serial2/0
C    20.0.0.0/8 is directly connected, FastEthernet0/0
R    30.0.0.0/8 [120/1] via 50.0.0.2, 00:00:25, Serial3/0
C    40.0.0.0/8 is directly connected, Serial2/0
C    50.0.0.0/8 is directly connected, Serial3/0
Router#

```

```
* UNAUTHORIZED COMMUNICATION OR COMPUTER USE, OR SPREADING OF VIRUSES IS PROHIBITED  
Router#config t  
Enter configuration commands, one per line. End with CNTL/Z.  
Router(config)#router rip  
Router(config-router)#network 10.0.0.0  
Router(config-router)#network 40.0.0.0  
Router(config-router)#network 50.0.0.0  
Router(config-router)#exit  
Router(config)#exit  
Router#  
%SYS-5-CONFIG_I: Configured from console by console  
s  
Router#show ip route  
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP  
      D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area  
      N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2  
      E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP  
      i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area  
      * - candidate default, U - per-user static route, o - ODR  
      P - periodic downloaded static route  
  
Gateway of last resort is not set  
  
C    10.0.0.0/8 is directly connected, FastEthernet0/0  
R    20.0.0.0/8 [120/1] via 40.0.0.2, 00:00:23, Serial2/0  
R    30.0.0.0/8 [120/2] via 40.0.0.2, 00:00:15, Serial2/0  
C    40.0.0.0/8 is directly connected, Serial2/0  
R    50.0.0.0/8 [120/1] via 40.0.0.2, 00:00:23, Serial2/0  
Router#
```

iv. Observation:

classmate
Date _____
Page _____

- b,c are connected successfully at port 1024
→ port 1024 is free
→ simulate the packet transmission from one device to another device.

OBSERVATION:

→ Packet transmission from source to destination is successful.

→ IP address are established.

→ TTL originally starts with a value 255

→ TTL decrements at each hop during packet transmission.

13.28.18 2019 8:30 AM 2019-08-13 10:30 AM

X/1/2X

2019-08-18 : 2019-08-18 9:00 AM 2019-08-18 9:00 AM

13.28.18 2019 8:30 AM 2019-08-13 10:30 AM

connection established with 192.168.1.10

after 7 sec (time of which passes with 5 sec)

of 192.168.1.10 and 192.168.1.10 connection established

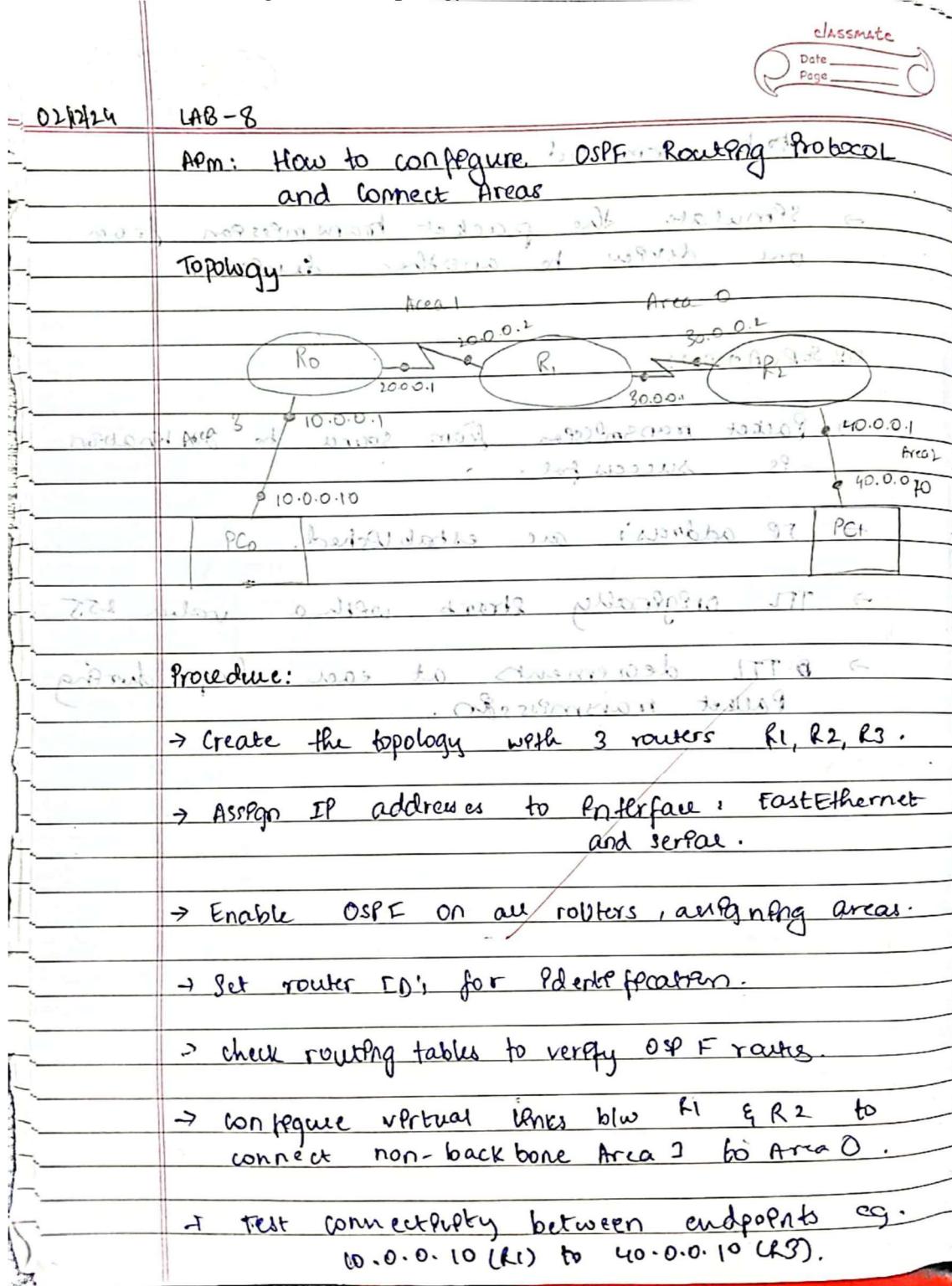
between 192.168.1.10 and 192.168.1.10

192.168.1.10 connected packet received from 192.168.1.10
(81) 01-0-0-0-0 of (14) 01-0-0-0-0

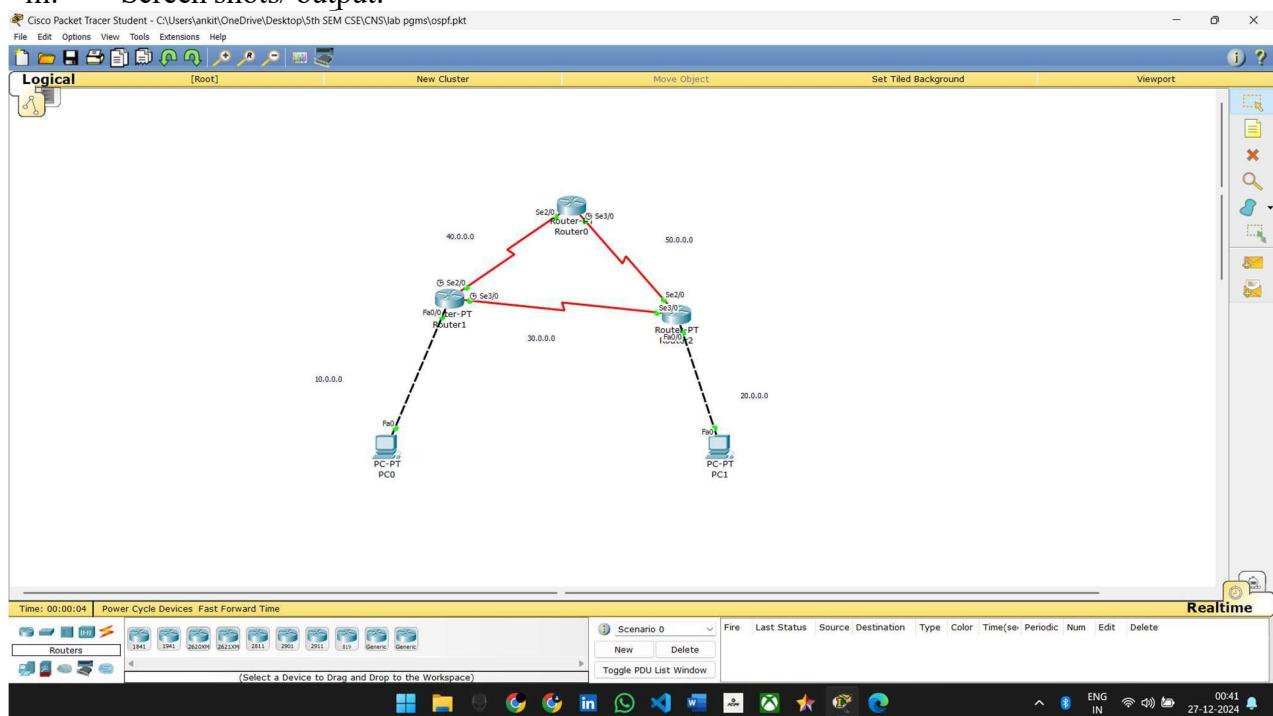
Program 6

i. Aim of the program: Configure OSPF routing protocol

ii. Procedure along with the topology:



iii. Screen shots/ output:



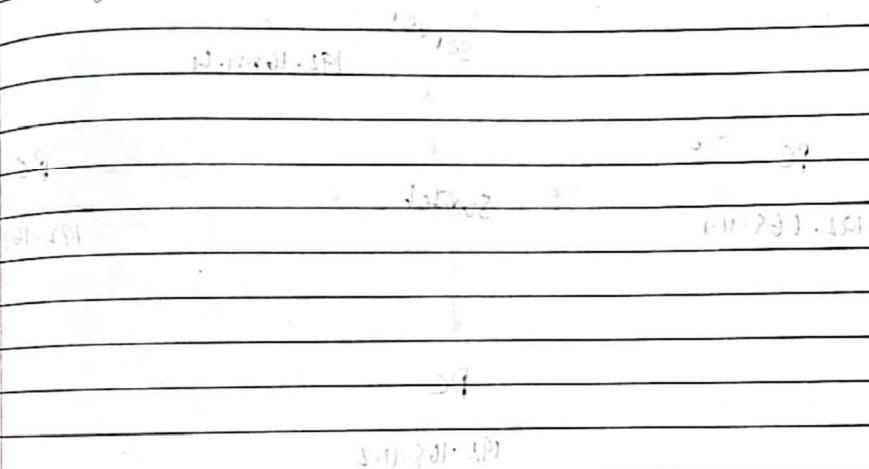
iv. Observation:

CLASSMATE
Date _____
Page _____
10/10/2019

Observation: ~~Information about OSPF~~

→ OSPF routes are dynamically learned, ensuring connectivity between all areas, including virtual links.

8/10/2019



→ output of show ip route command on R2 is as follows:

Output of show ip route command on R2
Shows all active routes in the database

→ Output of show ip route command on R3
Shows all active routes in the database

→ Output of show ip route command on R1
Shows all active routes in the database

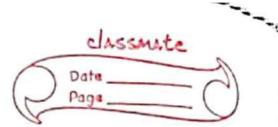
Program 7

i. Aim of the program: Demonstrate the TTL/ Life of a Packet

ii. Procedure along with the topology:

22/11/24

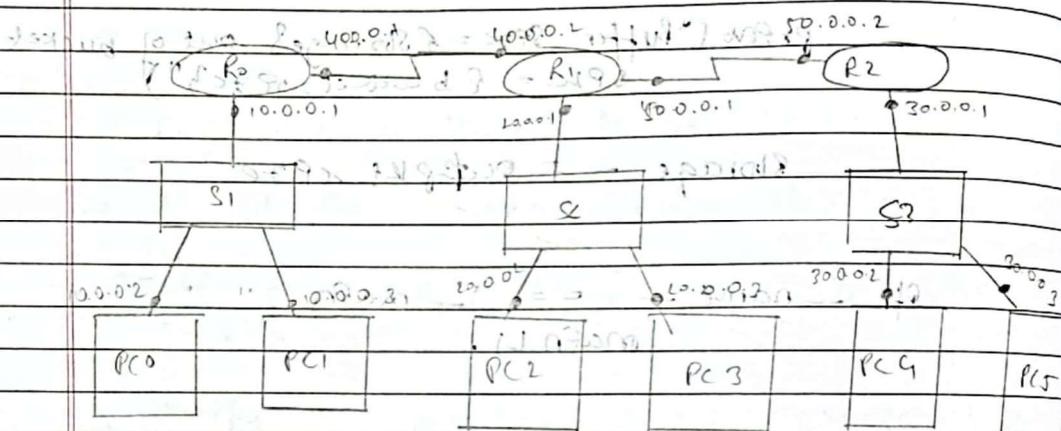
LAB - 7



Aim: Demonstrate RIP (Routing Information Protocol)

17/11/24

Topology:



Procedure:

1. Configure the routers and their fast Ethernet ports.

2. Configure the serial ports as shown in the figure.

3. Configure the gateway on end devices.

4. Open the CLI of each router, open config terminal and type 'router rip' after that type all the networks.

5. Configure the gateway on end devices.

6. Open the CLI of each router, open config terminal and type 'router rip' after that type all the networks.

a, b, c are connected in a ring at node 1, 2, 3

simulate the packet transmission from one device to another device.

OBSERVATION:

→ Packet transmission from source to destination is successful.

→ IP address are established.

→ TTL originally starts with a value 255

→ TTL decrements at each hop during packet transmission.

✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

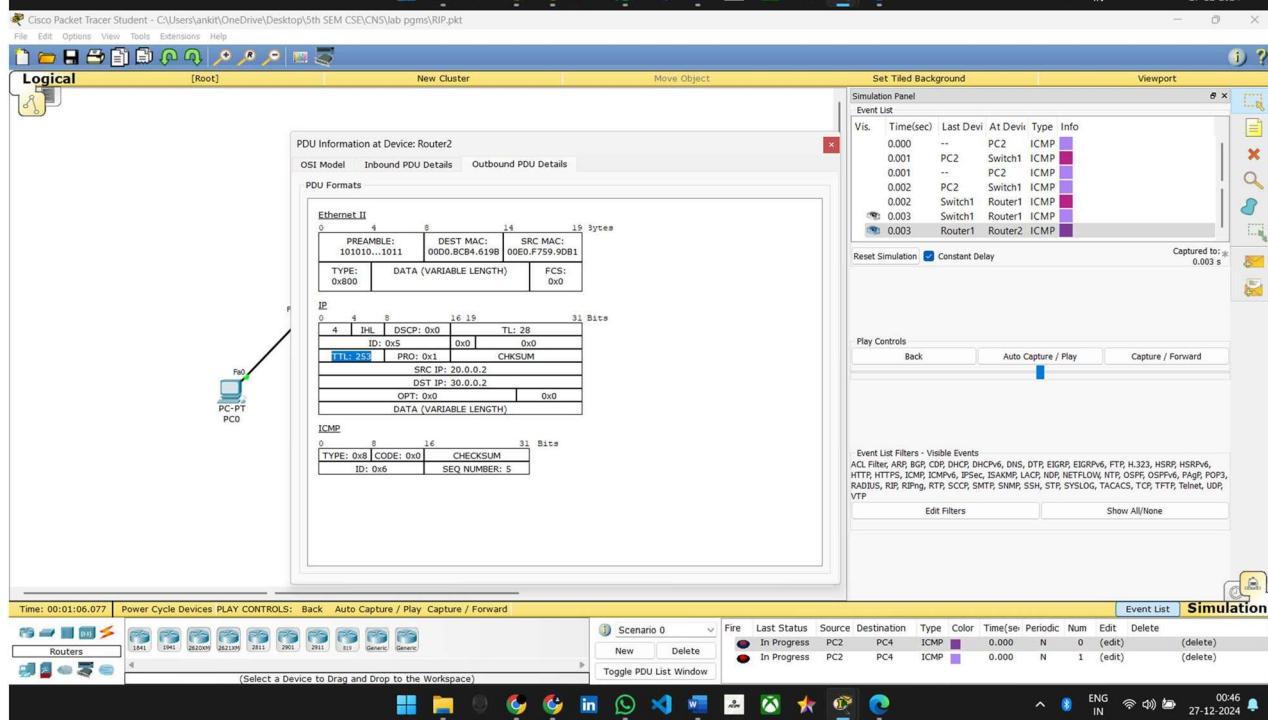
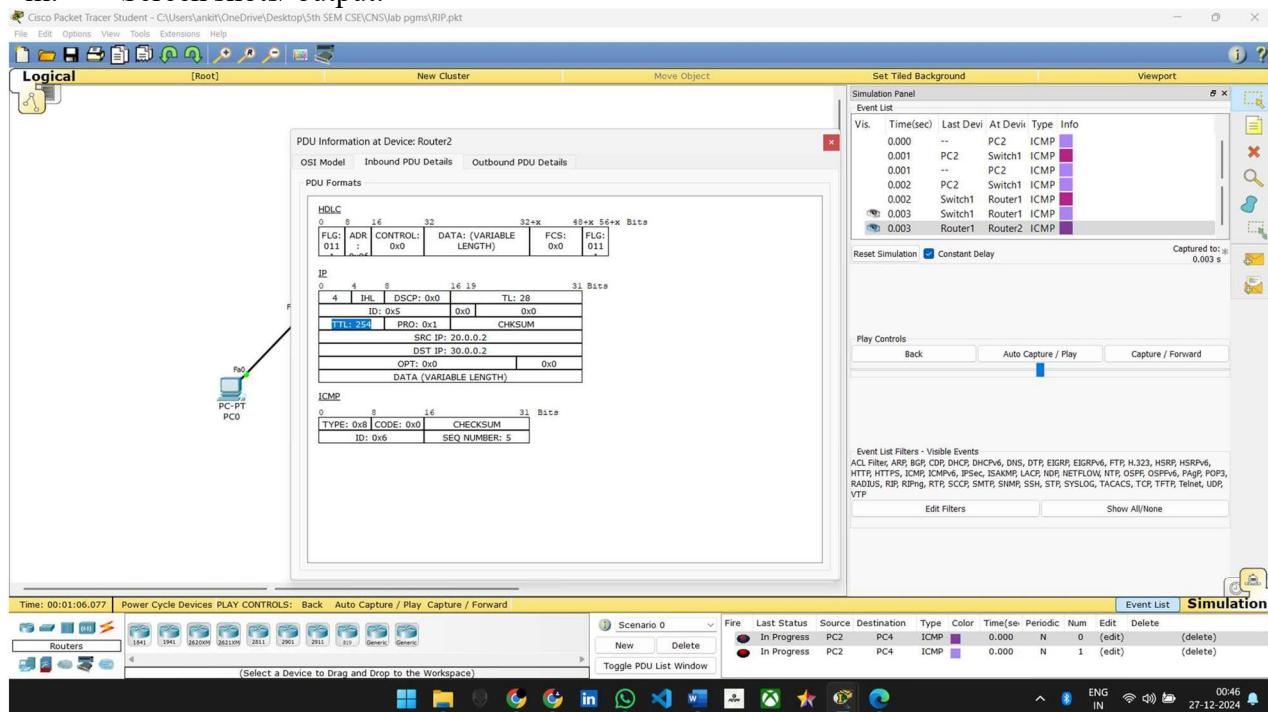
✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

✓ ~~After 8 steps, TTL becomes 0. So packet is lost.~~

iii. Screen shots/ output:



Observation:



- simulate the packet transmission from one device to another device.

OBSERVATION:

→ Packet transmission from source to destination is successful.

→ IP addresses are established.

→ TTL originally starts with a value 255

→ TTL decrements at each hop during packet transmission.

→ TTL is 255 initially and decreases to 0 after 5 hops.

→ TTL is 255 initially and decreases to 0 after 5 hops.

→ TTL is 255 initially and decreases to 0 after 5 hops.

→ TTL is 255 initially and decreases to 0 after 5 hops.

→ TTL is 255 initially and decreases to 0 after 5 hops.

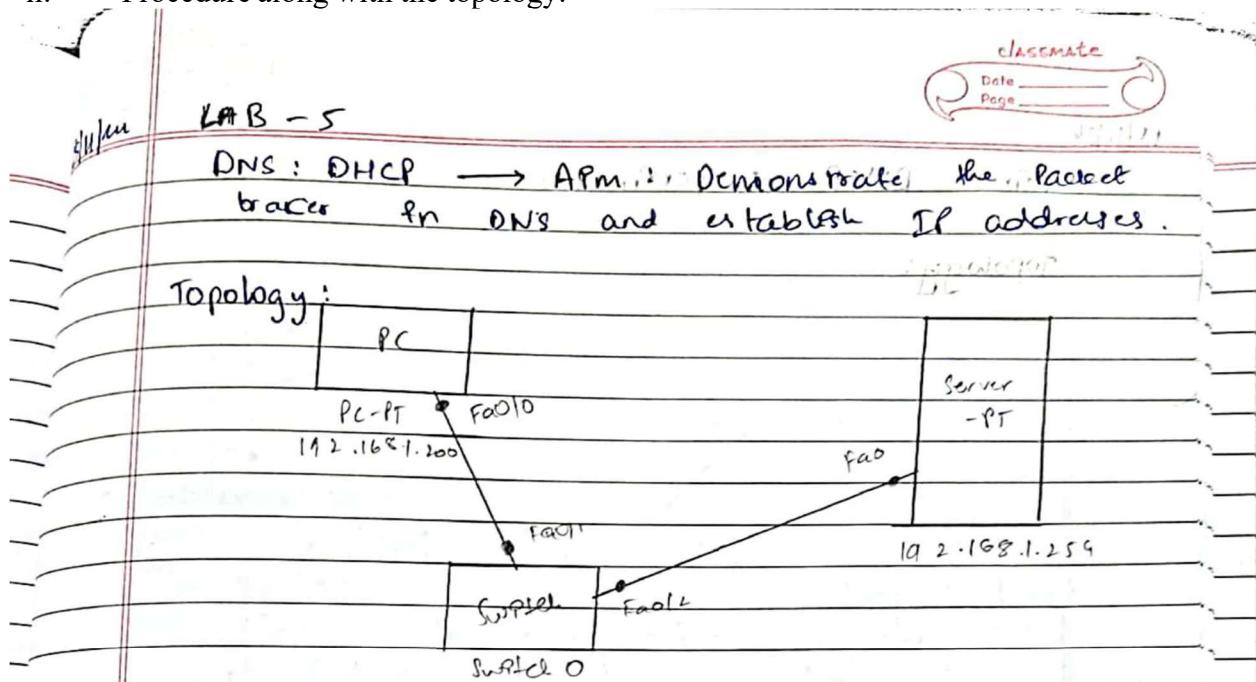
→ TTL is 255 initially and decreases to 0 after 5 hops.

→ TTL is 255 initially and decreases to 0 after 5 hops.

Program 8

i. Aim of the program: Configure Web Server, DNS within a LAN.

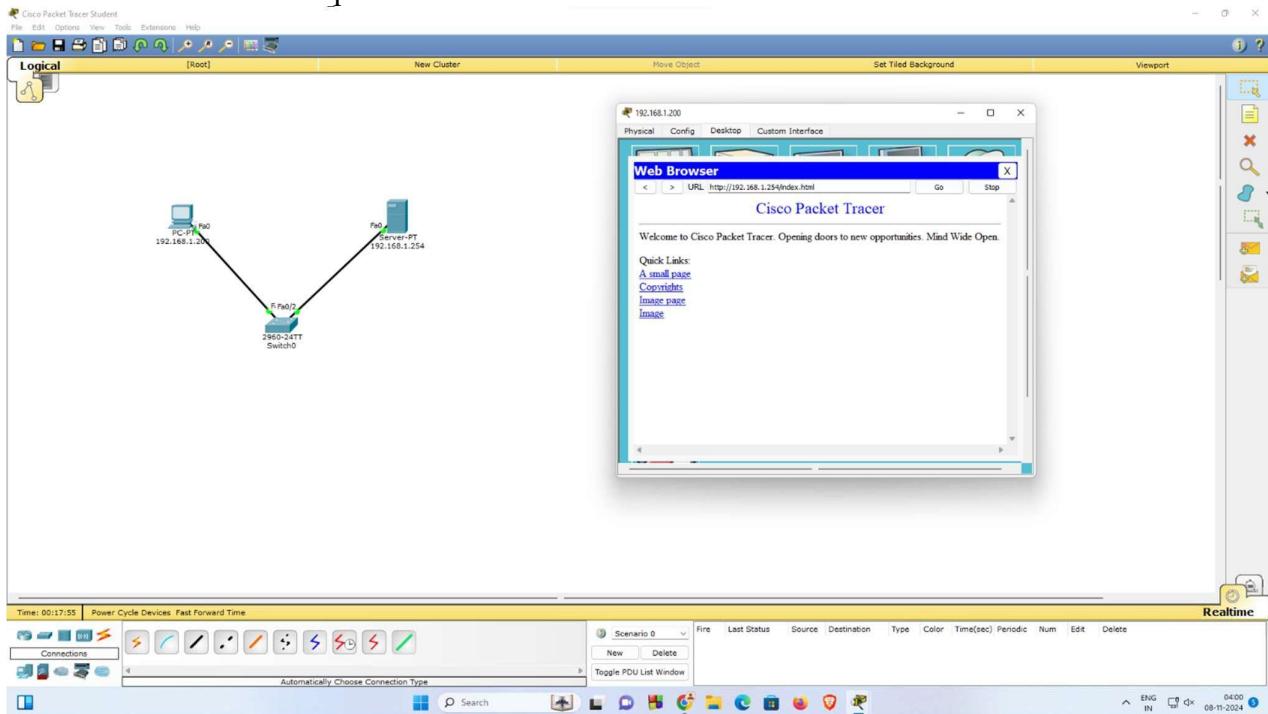
ii. Procedure along with the topology:



Procedure:

- Enable all connection between the PC, Switch and server and make the necessary connection.
- Establish IP address of all components.
- Assigning IP address to the server. (IP address 192.168.1.259)
- Open the web browser box and type the server IP address. (192.168.1.259)
- Web page must be visible.

iii. Screen shots/ output:

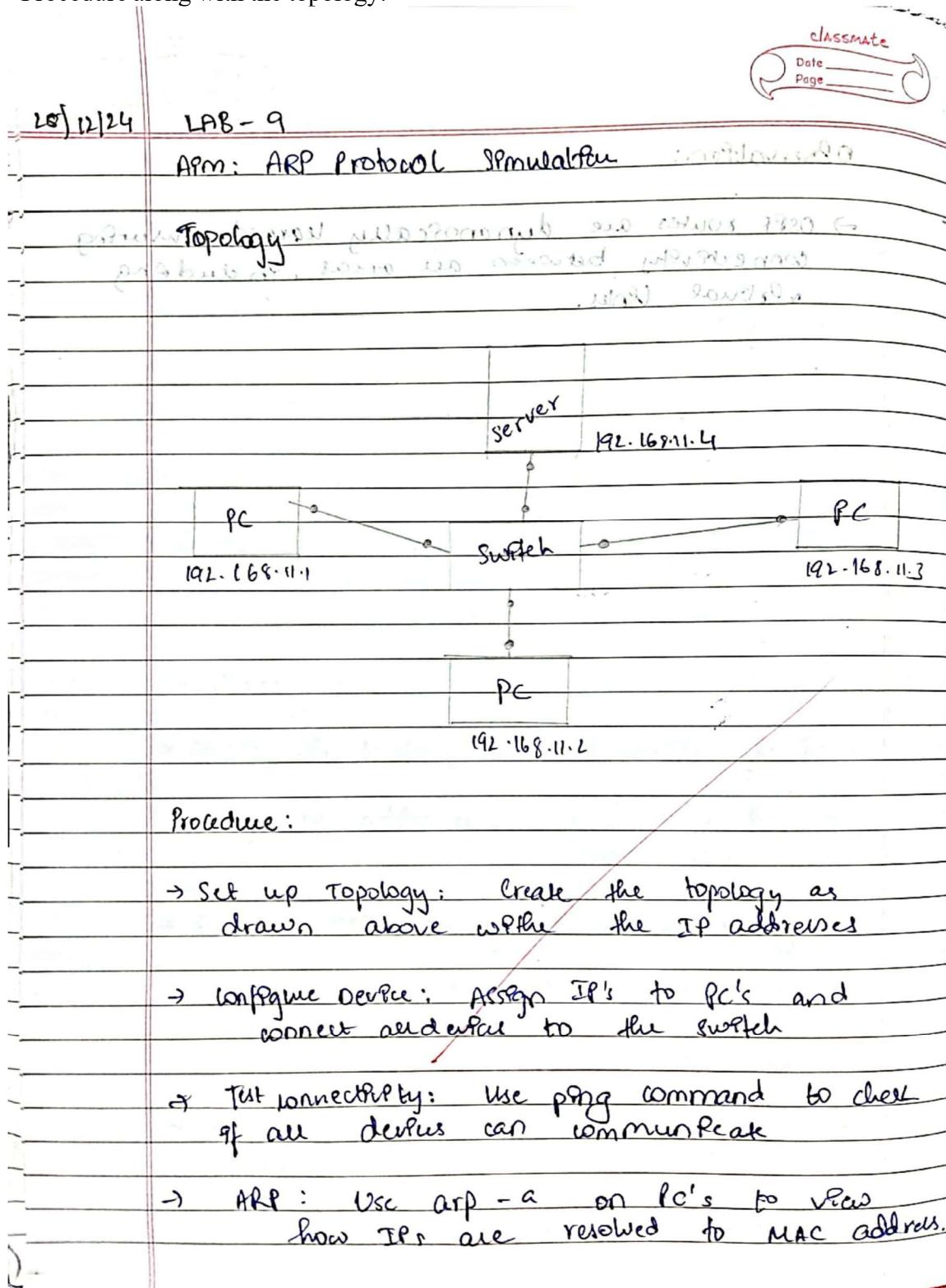


iv. Observation:

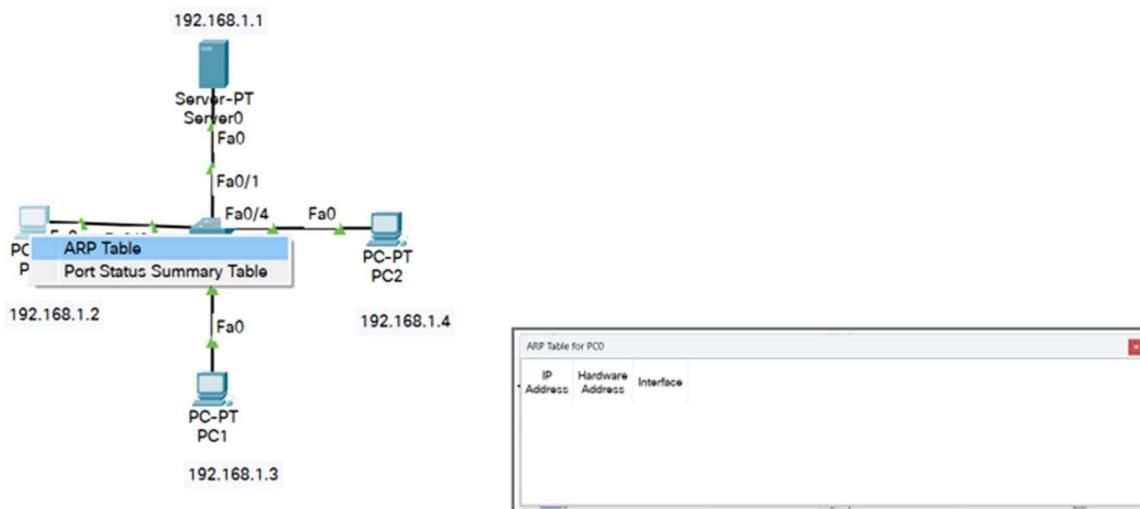
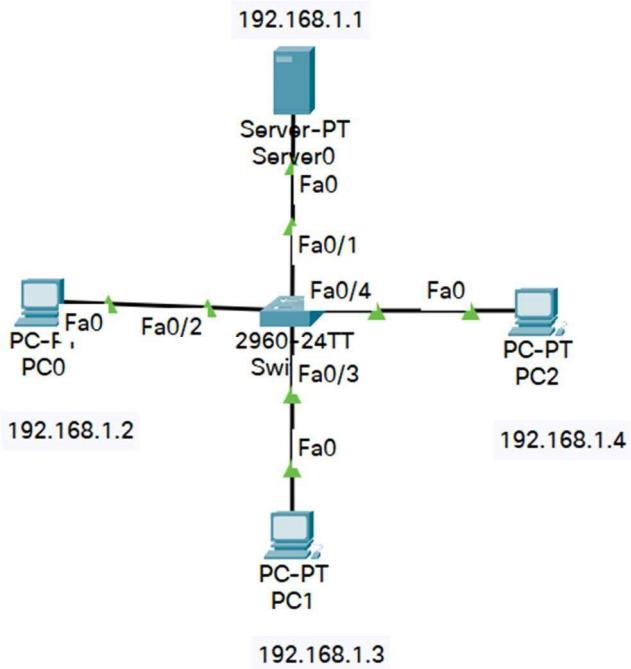
- classmate
Date _____
Page _____
P-24 10/10/17
- a, b, c are connected in a linear fashion.
- Simulate the packet transmission from one device to another device.
- OBSERVATION:**
- Packet transmission from source to destination is successful.
 - IP addresses are established.
 - TTL originally starts with a value 255.
 - TTL decrements at each hop during packet transmission.
- ~~1. 22.22.22.22 is the gateway with address
& 128.128.128.128~~
- ~~2. 22.22.22.22 is the gateway of network 9 & 10 & 11 & 12~~
- ~~3. 22.22.22.22 is the gateway of network 13 & 14 & 15 & 16~~
- ~~4. 22.22.22.22 is the gateway of network 17 & 18 & 19 & 20~~
- ~~5. 22.22.22.22 is the gateway of network 21 & 22 & 23 & 24~~
- ~~6. 22.22.22.22 is the gateway of network 25 & 26 & 27 & 28~~
- ~~7. 22.22.22.22 is the gateway of network 29 & 30 & 31 & 32~~
- ~~8. 22.22.22.22 is the gateway of network 33 & 34 & 35 & 36~~
- ~~9. 22.22.22.22 is the gateway of network 37 & 38 & 39 & 40~~
- ~~10. 22.22.22.22 is the gateway of network 41 & 42 & 43 & 44~~
- ~~11. 22.22.22.22 is the gateway of network 45 & 46 & 47 & 48~~
- ~~12. 22.22.22.22 is the gateway of network 49 & 50 & 51 & 52~~
- ~~13. 22.22.22.22 is the gateway of network 53 & 54 & 55 & 56~~
- ~~14. 22.22.22.22 is the gateway of network 57 & 58 & 59 & 60~~
- ~~15. 22.22.22.22 is the gateway of network 61 & 62 & 63 & 64~~
- ~~16. 22.22.22.22 is the gateway of network 65 & 66 & 67 & 68~~
- ~~17. 22.22.22.22 is the gateway of network 69 & 70 & 71 & 72~~
- ~~18. 22.22.22.22 is the gateway of network 73 & 74 & 75 & 76~~
- ~~19. 22.22.22.22 is the gateway of network 77 & 78 & 79 & 80~~
- ~~20. 22.22.22.22 is the gateway of network 81 & 82 & 83 & 84~~

Program 9

- i. Aim of the program: To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)
- ii. Procedure along with the topology:

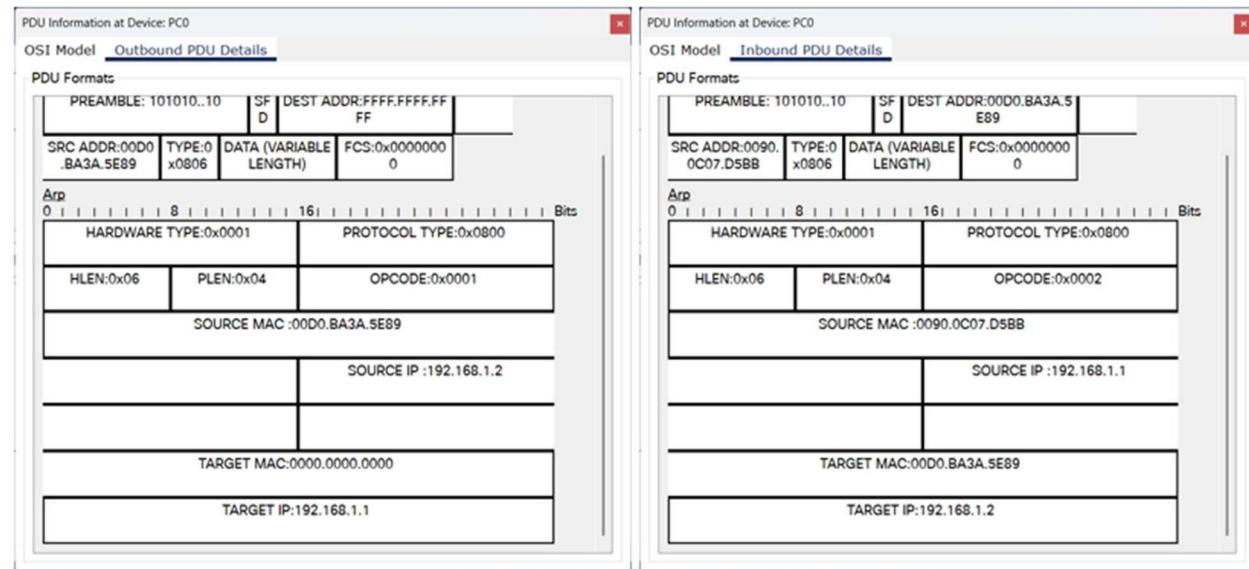


iii. Screen shots/ output:



```
Packet Tracer PC Command Line 1.0  
C:\>arp -a  
No ARP Entries Found  
C:\>
```

Pinging in Simulation Mode

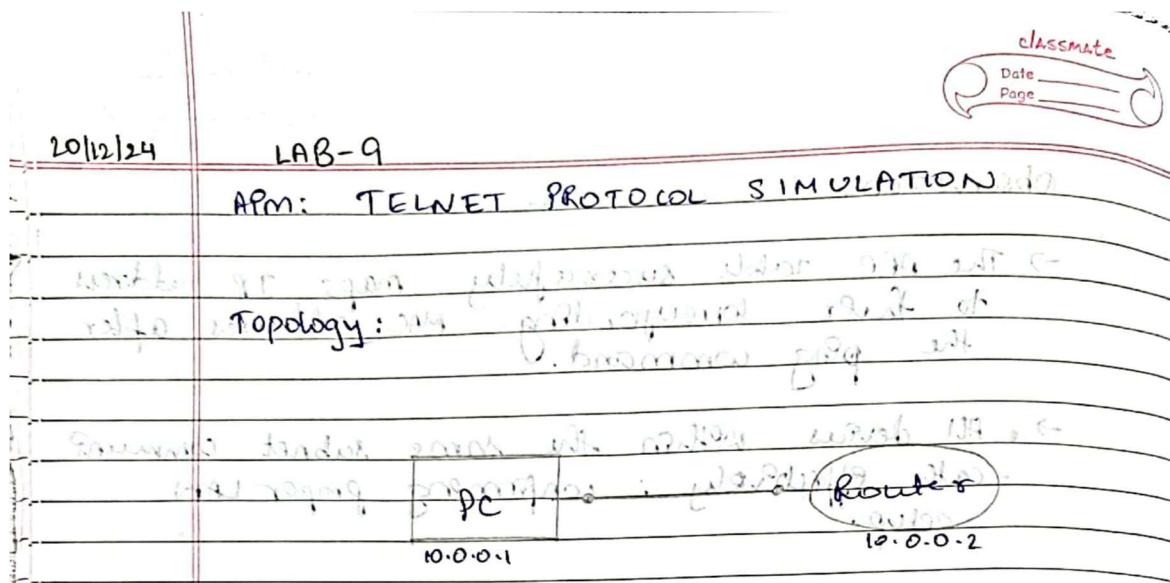


iv. Observation:

- classmate
Date _____
Page _____
- observation
- The ARP table successfully maps IP address to their corresponding MAC addresses after the ping command.
 - All devices within the same subnet communicate effectively, confirming proper LAN setup.
- add records in broadcast with MAC →
broadcast has worked if assigned to 009
TOS or subnet IP assigned →
bcast broadcast frame 192.168.6.
Broadcasting
bcast frame broadcast → 09 00 00 →
means 192.168.6.9
over 255 bits of broadcast will exist →
duration
frame propagation
width of 1000 bytes would help to prevent loss of packets with collision

Program 10

- i. Aim of the program: To understand the operation of TELNET by accessing the router in server room from a PC in IT office.
- ii. Procedure along with the topology:



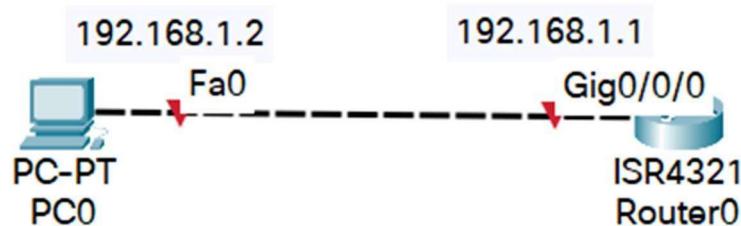
Procedure :

- Create the topology as shown above
- Configure IP address and gateway for PC
- Configure the router in CLI
- Set the secret password and password
- Go to PC - command prompt and ping the router .
- Type the password to establish remote control.

Observation :

- Telnet allows remote access to device over the network by establishing

iii. Screen shots/ output:

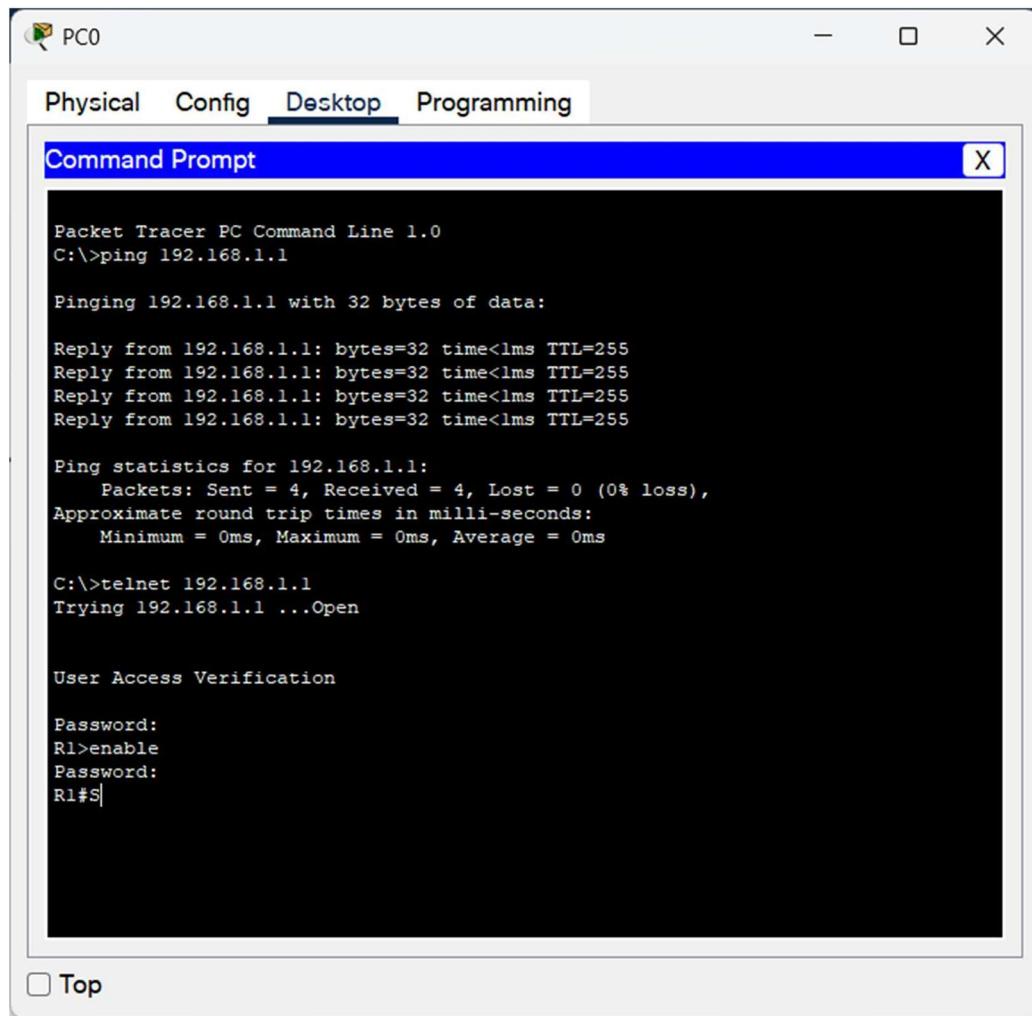


```
Router>enable
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#hostname R1
R1(config)#enable secret hello
R1(config)#interface g0/0/0
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#no shutdown

R1(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0/0, changed state to up

R1(config-if)#line vty 0 5
R1(config-line)#login
% Login disabled on line 2, until 'password' is set
% Login disabled on line 3, until 'password' is set
% Login disabled on line 4, until 'password' is set
% Login disabled on line 5, until 'password' is set
% Login disabled on line 6, until 'password' is set
% Login disabled on line 7, until 'password' is set
R1(config-line)#password pass
R1(config-line)#end
R1#
%SYS-5-CONFIG_I: Configured from console by console

R1#wr
Building configuration...
[OK]
R1#|
```



Top

iv. Observation:

classmate

Data
Page

a command-line interface

→ Successful login to the Remote device. Confirms proper configuration of the telnet protocol and network connectivity.

1. 192.168.1.101 192.168.1.101

2. 192.168.1.101 192.168.1.101

3. 192.168.1.101 192.168.1.101

4. 192.168.1.101 192.168.1.101

5. 192.168.1.101 192.168.1.101

6. 192.168.1.101 192.168.1.101

7. 192.168.1.101 192.168.1.101

8. 192.168.1.101 192.168.1.101

9. 192.168.1.101 192.168.1.101

10. 192.168.1.101 192.168.1.101

11. 192.168.1.101 192.168.1.101

12. 192.168.1.101 192.168.1.101

13. 192.168.1.101 192.168.1.101

14. 192.168.1.101 192.168.1.101

15. 192.168.1.101 192.168.1.101

16. 192.168.1.101 192.168.1.101

17. 192.168.1.101 192.168.1.101

18. 192.168.1.101 192.168.1.101

19. 192.168.1.101 192.168.1.101

20. 192.168.1.101 192.168.1.101

21. 192.168.1.101 192.168.1.101

22. 192.168.1.101 192.168.1.101

23. 192.168.1.101 192.168.1.101

24. 192.168.1.101 192.168.1.101

25. 192.168.1.101 192.168.1.101

26. 192.168.1.101 192.168.1.101

27. 192.168.1.101 192.168.1.101

28. 192.168.1.101 192.168.1.101

29. 192.168.1.101 192.168.1.101

30. 192.168.1.101 192.168.1.101

31. 192.168.1.101 192.168.1.101

32. 192.168.1.101 192.168.1.101

33. 192.168.1.101 192.168.1.101

34. 192.168.1.101 192.168.1.101

35. 192.168.1.101 192.168.1.101

36. 192.168.1.101 192.168.1.101

37. 192.168.1.101 192.168.1.101

38. 192.168.1.101 192.168.1.101

39. 192.168.1.101 192.168.1.101

40. 192.168.1.101 192.168.1.101

41. 192.168.1.101 192.168.1.101

42. 192.168.1.101 192.168.1.101

43. 192.168.1.101 192.168.1.101

44. 192.168.1.101 192.168.1.101

45. 192.168.1.101 192.168.1.101

46. 192.168.1.101 192.168.1.101

47. 192.168.1.101 192.168.1.101

48. 192.168.1.101 192.168.1.101

49. 192.168.1.101 192.168.1.101

50. 192.168.1.101 192.168.1.101

51. 192.168.1.101 192.168.1.101

52. 192.168.1.101 192.168.1.101

53. 192.168.1.101 192.168.1.101

54. 192.168.1.101 192.168.1.101

55. 192.168.1.101 192.168.1.101

56. 192.168.1.101 192.168.1.101

57. 192.168.1.101 192.168.1.101

58. 192.168.1.101 192.168.1.101

59. 192.168.1.101 192.168.1.101

60. 192.168.1.101 192.168.1.101

61. 192.168.1.101 192.168.1.101

62. 192.168.1.101 192.168.1.101

63. 192.168.1.101 192.168.1.101

64. 192.168.1.101 192.168.1.101

65. 192.168.1.101 192.168.1.101

66. 192.168.1.101 192.168.1.101

67. 192.168.1.101 192.168.1.101

68. 192.168.1.101 192.168.1.101

69. 192.168.1.101 192.168.1.101

70. 192.168.1.101 192.168.1.101

71. 192.168.1.101 192.168.1.101

72. 192.168.1.101 192.168.1.101

73. 192.168.1.101 192.168.1.101

74. 192.168.1.101 192.168.1.101

75. 192.168.1.101 192.168.1.101

76. 192.168.1.101 192.168.1.101

77. 192.168.1.101 192.168.1.101

78. 192.168.1.101 192.168.1.101

79. 192.168.1.101 192.168.1.101

80. 192.168.1.101 192.168.1.101

81. 192.168.1.101 192.168.1.101

82. 192.168.1.101 192.168.1.101

83. 192.168.1.101 192.168.1.101

84. 192.168.1.101 192.168.1.101

85. 192.168.1.101 192.168.1.101

86. 192.168.1.101 192.168.1.101

87. 192.168.1.101 192.168.1.101

88. 192.168.1.101 192.168.1.101

89. 192.168.1.101 192.168.1.101

90. 192.168.1.101 192.168.1.101

91. 192.168.1.101 192.168.1.101

92. 192.168.1.101 192.168.1.101

93. 192.168.1.101 192.168.1.101

94. 192.168.1.101 192.168.1.101

95. 192.168.1.101 192.168.1.101

96. 192.168.1.101 192.168.1.101

97. 192.168.1.101 192.168.1.101

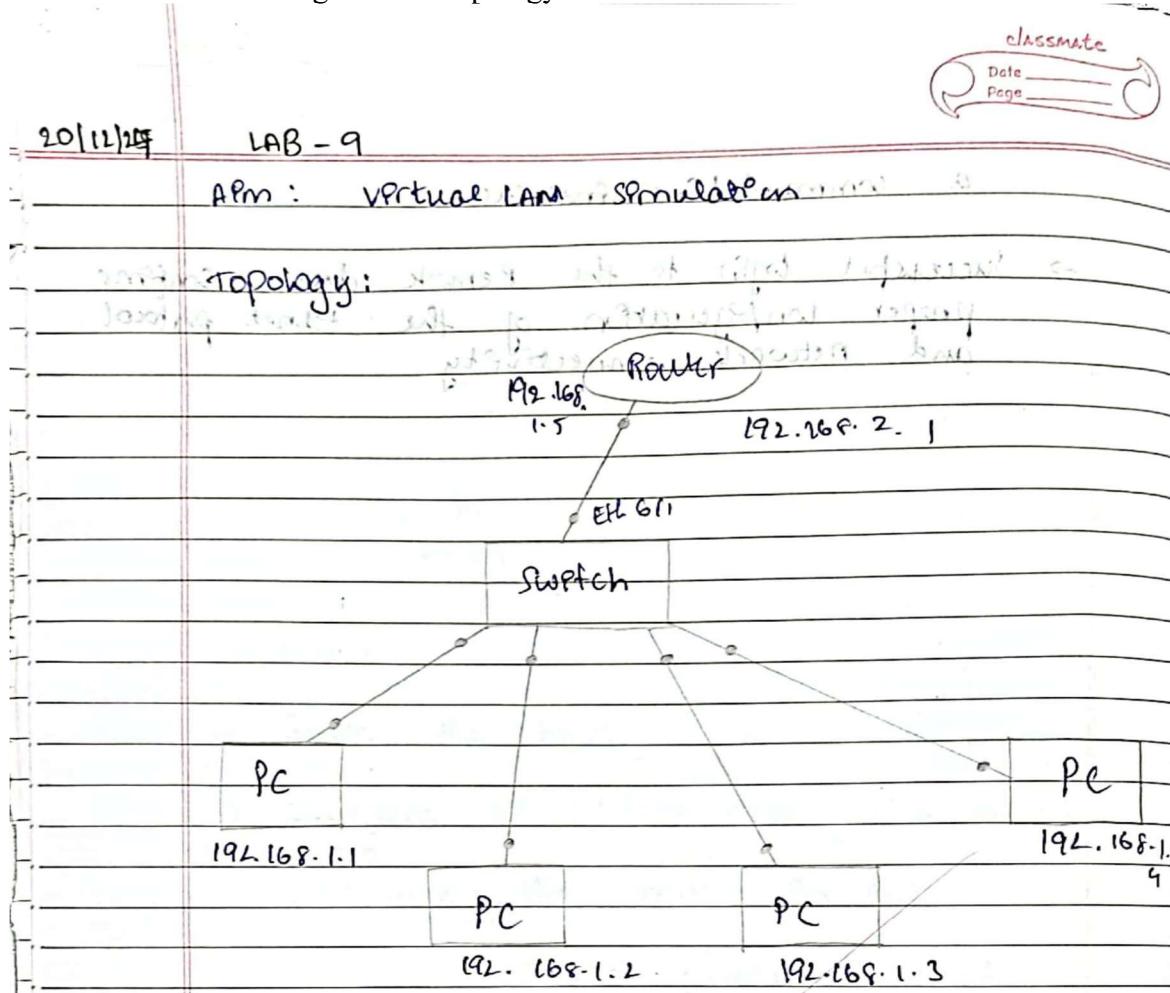
98. 192.168.1.101 192.168.1.101

99. 192.168.1.101 192.168.1.101

100. 192.168.1.101 192.168.1.101

Program 11

- i. Aim of the program: To construct a VLAN and make the PC's communicate among a VLAN
- ii. Procedure along with the topology:



Procedure :

→ Create the topology as shown above . Set the switch interface connected to the router as a trunk .

→ Enable VLAN Tagging allows frames from different VLANs to pass through to trunk link .

→ In the routers' VLAN database , define VLAN

number and name i.e. Ans0gr1 a sub-interface for the VLAN with dot1q encapsulation and an IP address.

→ Activate VLAN by using no shut command.
Save it and verify the configuration.

Observation :

- VLAN tagging ensures proper data segregation and communication between devices in different VLANs through the trunk link.
- Successful communication is established when the router and switch are configured correctly with VLAN tagging.

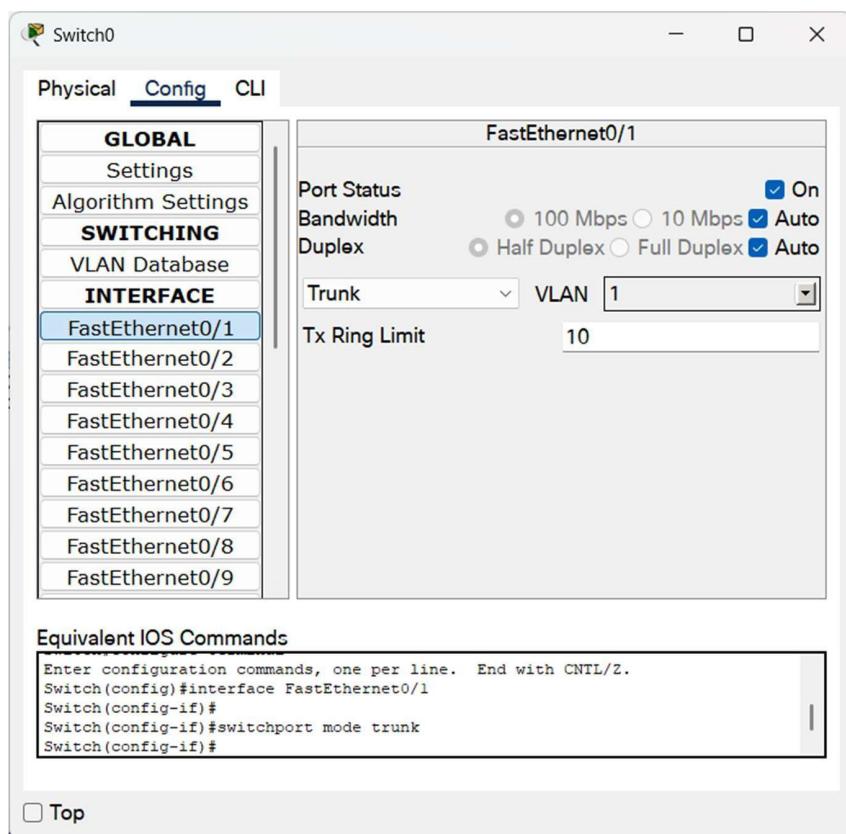
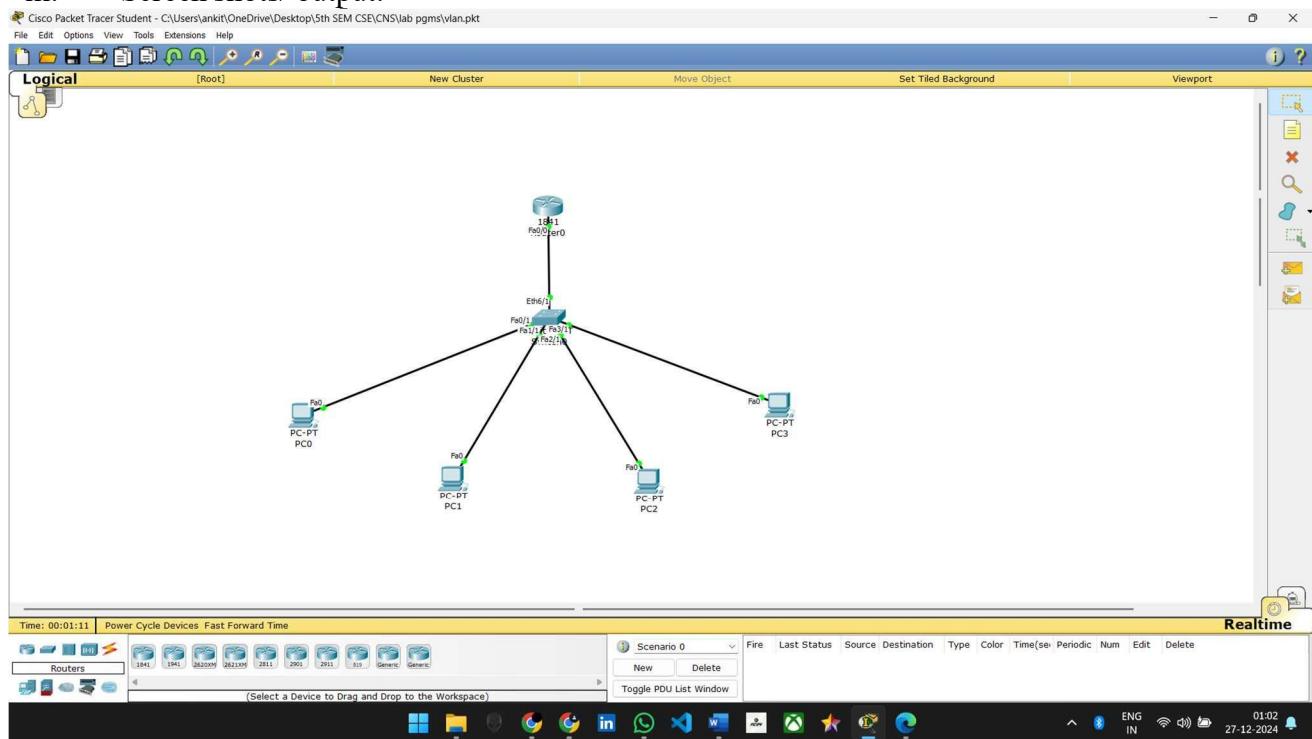
Port 1/0/23 S-0-0-23 1.3.0.0/16
Port 1/0/24 S-0-0-24 1.3.0.1/24

Assigned VLAN 23. Router Configuration (Ans0gr1)
↳ enable mode

↳ config mode
↳ interface port 1/0/23
↳ ip address 1.3.0.2 255.255.255.0
↳ end mode

↳ config mode
↳ interface port 1/0/24
↳ ip address 1.3.0.1 255.255.255.0
↳ end mode

iii. Screen shots/ output:



Switch0

Physical Config CLI

GLOBAL	
Settings	
Algorithm Settings	
SWITCHING	
VLAN Database	
INTERFACE	
FastEthernet0/1	
FastEthernet0/2	
FastEthernet0/3	
FastEthernet0/4	
FastEthernet0/5	
FastEthernet0/6	
FastEthernet0/7	
FastEthernet0/8	
FastEthernet0/9	

FastEthernet0/4

Port Status: On
 Bandwidth: 100 Mbps 10 Mbps Auto
 Duplex: Half Duplex Full Duplex Auto

Access: VLAN

Tx Ring Limit:

FastEthernet0/5

Port Status: On
 Bandwidth: 100 Mbps 10 Mbps Auto
 Duplex: Half Duplex Full Duplex Auto

Access: VLAN

Tx Ring Limit:

- 1:default
- 2:VLAN
- 1002:fddi-default

Top

Switch0

Physical Config CLI

GLOBAL	
Settings	
Algorithm Settings	
SWITCHING	
VLAN Database	
INTERFACE	
FastEthernet0/1	
FastEthernet0/2	
FastEthernet0/3	
FastEthernet0/4	
FastEthernet0/5	
FastEthernet0/6	
FastEthernet0/7	
FastEthernet0/8	
FastEthernet0/9	

Equivalent IOS Commands

```
Switch(config)#interface FastEthernet0/4
Switch(config-if)#
Switch(config-if)#
Switch(config-if)#switchport access vlan 2
Switch(config-if)#

```

Equivalent IOS Commands

```
Switch(config)#interface FastEthernet0/5
Switch(config-if)#
Switch(config-if)#
Switch(config-if)#switchport access vlan 2
Switch(config-if)#

```

Top

Router0

Physical Config CLI

IOS Command Line Interface

```
Router>enable
Router#config t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#interface Fa0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed
state to up

Router(config-if)#exit
Router(config)#interface Fa0/0.1
Router(config-subif)#
%LINK-5-CHANGED: Interface FastEthernet0/0.1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0.1, changed
state to up

Router(config-subif)#encapsulation dot1q 2
Router(config-subif)#ip address 20.0.0.1 255.0.0.0
Router(config-subif)#no shutdown
Router(config-subif)#exit
Router(config)#

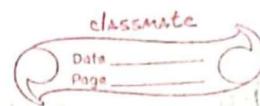
```

Ctrl+F6 to exit CLI focus

Copy **Paste**

Top

iv. Observation:



number and name via Ans Pgs: a sub-interface for the VLAN with dot1q encapsulation and an IP address.

→ Activate VLAN by using no shut command.
Save pt and verify the configuration.

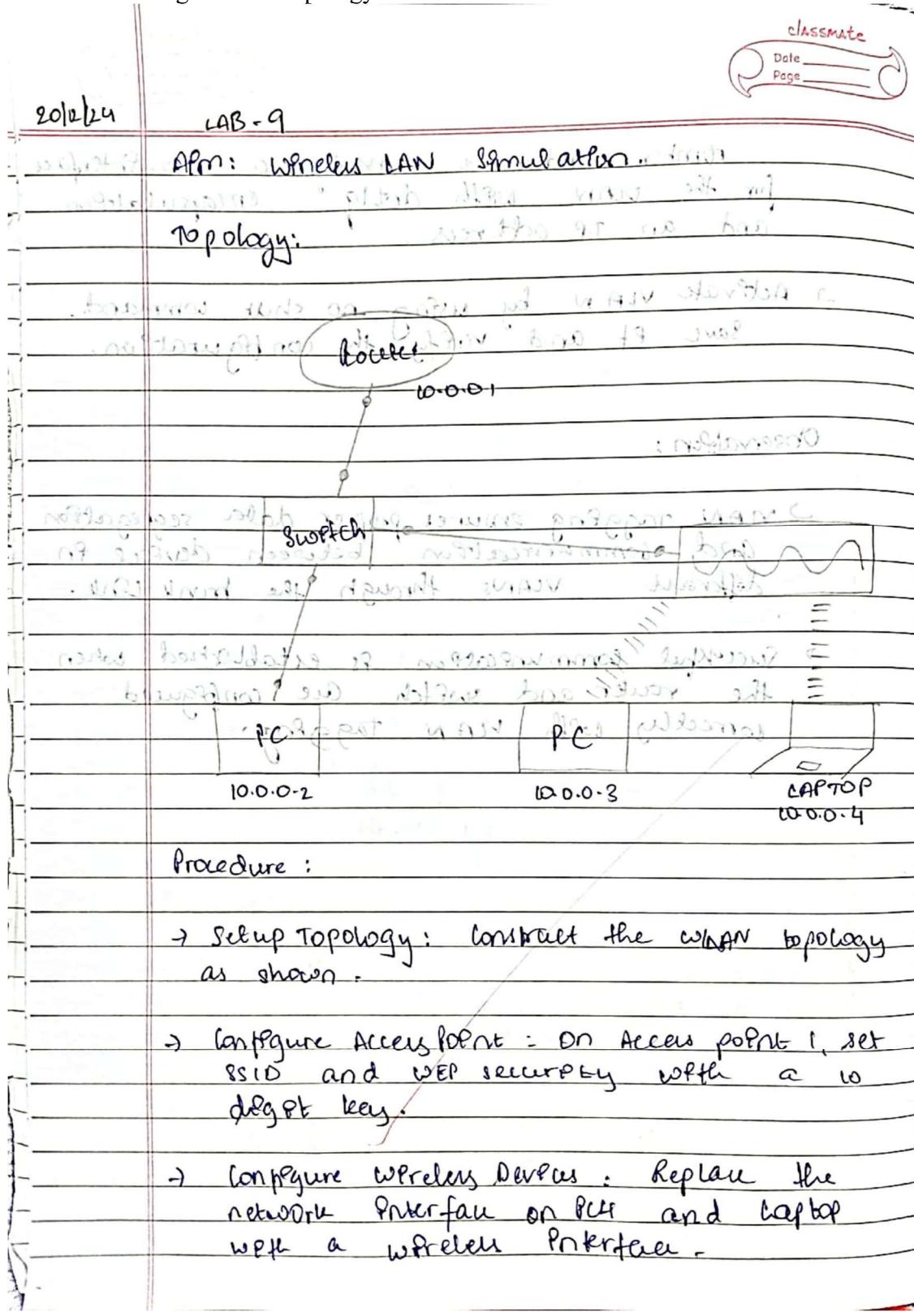
Observation:

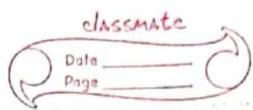
- VLAN tagging ensures proper data segregation and communication between devices in different VLANs through the trunk link.
- Successful communication is established when the router and switch are configured correctly with VLAN tagging.

1. 192.168.1.10
2. 192.168.1.11
3. 192.168.1.12
4. 192.168.1.13
5. 192.168.1.14
6. 192.168.1.15
7. 192.168.1.16
8. 192.168.1.17
9. 192.168.1.18
10. 192.168.1.19
11. 192.168.1.20
12. 192.168.1.21
13. 192.168.1.22
14. 192.168.1.23
15. 192.168.1.24
16. 192.168.1.25
17. 192.168.1.26
18. 192.168.1.27
19. 192.168.1.28
20. 192.168.1.29
21. 192.168.1.30
22. 192.168.1.31
23. 192.168.1.32
24. 192.168.1.33
25. 192.168.1.34
26. 192.168.1.35
27. 192.168.1.36
28. 192.168.1.37
29. 192.168.1.38
30. 192.168.1.39
31. 192.168.1.40
32. 192.168.1.41
33. 192.168.1.42
34. 192.168.1.43
35. 192.168.1.44
36. 192.168.1.45
37. 192.168.1.46
38. 192.168.1.47
39. 192.168.1.48
40. 192.168.1.49
41. 192.168.1.50
42. 192.168.1.51
43. 192.168.1.52
44. 192.168.1.53
45. 192.168.1.54
46. 192.168.1.55
47. 192.168.1.56
48. 192.168.1.57
49. 192.168.1.58
50. 192.168.1.59
51. 192.168.1.60
52. 192.168.1.61
53. 192.168.1.62
54. 192.168.1.63
55. 192.168.1.64
56. 192.168.1.65
57. 192.168.1.66
58. 192.168.1.67
59. 192.168.1.68
60. 192.168.1.69
61. 192.168.1.70
62. 192.168.1.71
63. 192.168.1.72
64. 192.168.1.73
65. 192.168.1.74
66. 192.168.1.75
67. 192.168.1.76
68. 192.168.1.77
69. 192.168.1.78
70. 192.168.1.79
71. 192.168.1.80
72. 192.168.1.81
73. 192.168.1.82
74. 192.168.1.83
75. 192.168.1.84
76. 192.168.1.85
77. 192.168.1.86
78. 192.168.1.87
79. 192.168.1.88
80. 192.168.1.89
81. 192.168.1.90
82. 192.168.1.91
83. 192.168.1.92
84. 192.168.1.93
85. 192.168.1.94
86. 192.168.1.95
87. 192.168.1.96
88. 192.168.1.97
89. 192.168.1.98
90. 192.168.1.99
91. 192.168.1.100

Program 12

- i. Aim of the program: To construct a WLAN and make the nodes communicate wirelessly
- ii. Procedure along with the topology:



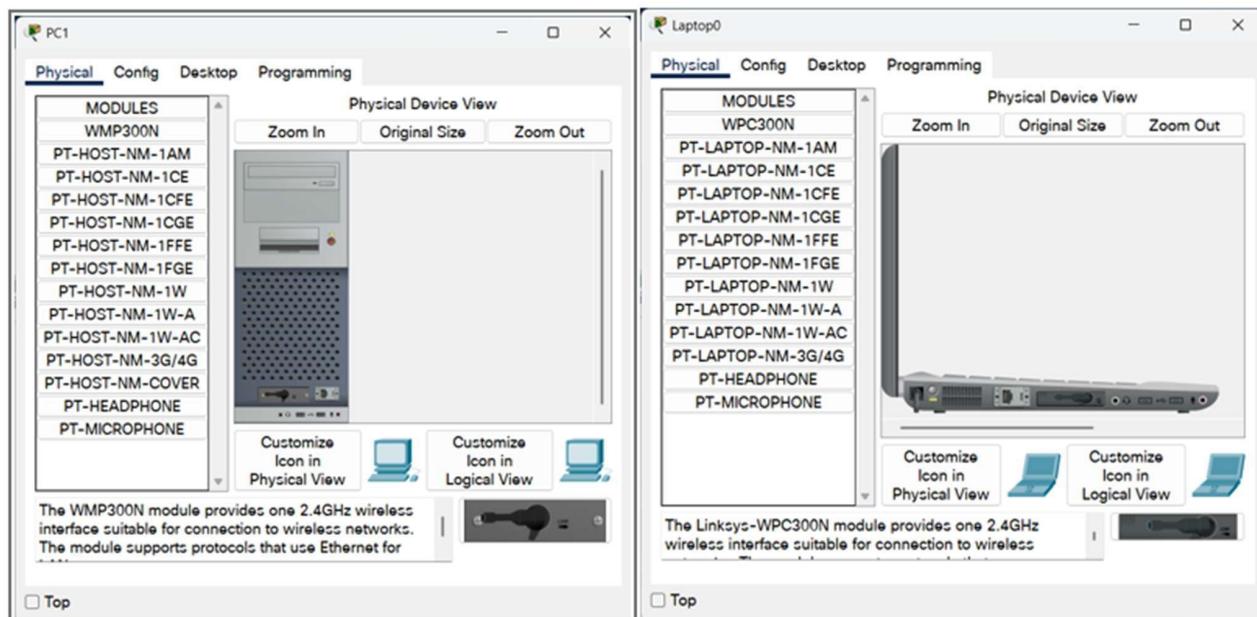
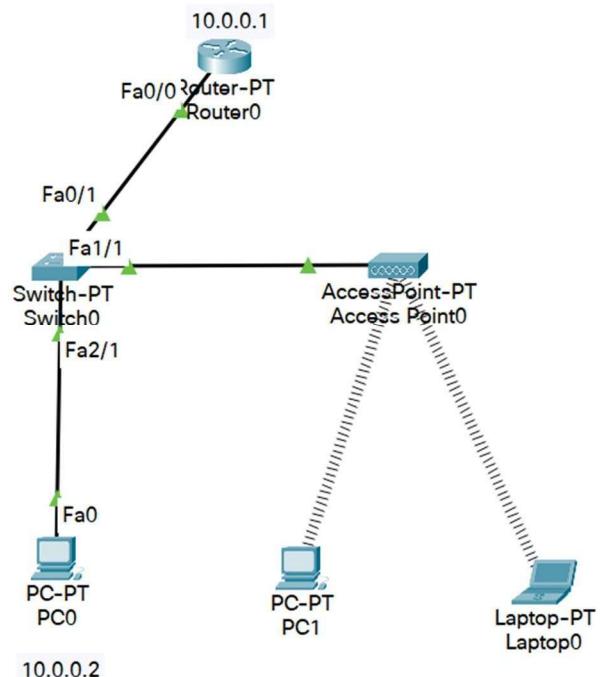


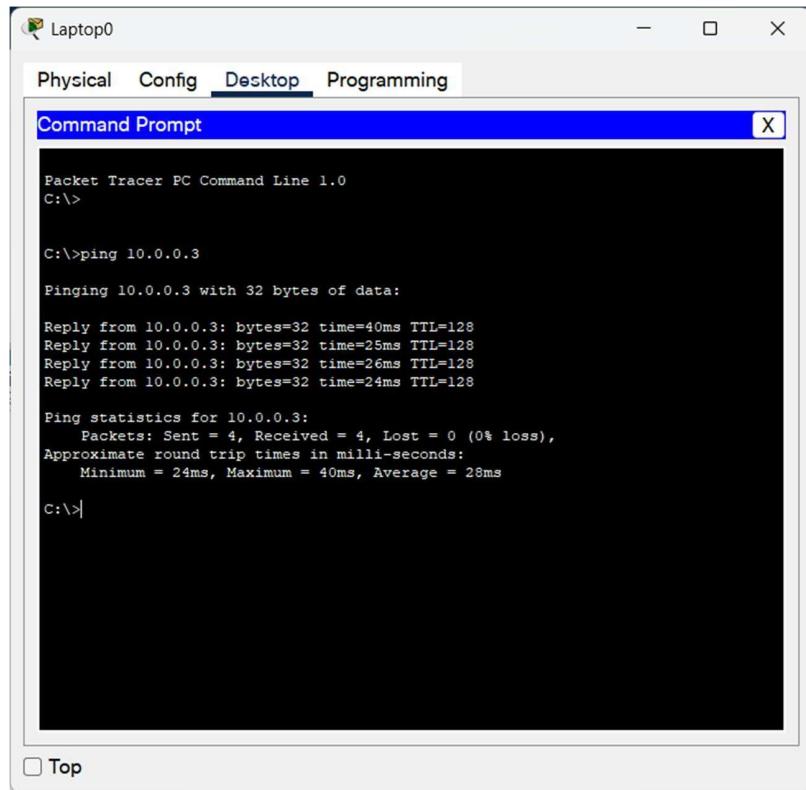
→ Pinging all devices within the network to verify communication.

Observation:

- Successful wireless communication occurs between all devices when properly configured with matching SSID.
- Devices respond to pings, confirming a fully functional WLAN setup.

iii. Screen shots/ output:





iv. Observation:

→ Pinging all devices within the network to verify communication.

Observation:

→ Successful wireless communication occurs between all devices when properly configured with matching SSID.

→ Devices respond to pings, confirming a fully functional WLAN setup.

CYCLE - 2

Program 1

- i. Aim of the program: Write a program for error detecting code using CRC-CCITT (16-bits).
- ii. Procedure:

Aim: CRC code and leaky bucket Algorithm
Implementation.

Algorithm : CRC-L-CCITT (16.bpts) algorithm

→ Start

→ Enter the message to be transmitted

→ Append the message with 16pt 0's.

→ XOR appended message and transmit pt.

→ Verify the message that is received is the same as the one sent

→ End

iii. Screen shots code:

```
def crc(input_msg, poly, mode):
    # Prepare the output message
    output_msg = input_msg
    if mode:
        output_msg += '0' * (len(poly) - 1)

    # Perform XOR on the message with the selected polynomial
    output_msg = list(output_msg)
    poly_len = len(poly)
    for i in range(len(input_msg)):
        if output_msg[i] == '1':
            for j in range(poly_len):
                if i + j < len(output_msg):
                    output_msg[i + j] = '0' if output_msg[i + j] == poly[j] else '1'

    # Check for errors and return 0 if error detected
    return '1' not in output_msg[-(poly_len-1):]

def main():
    poly = "1000100000100001"

    # Input the original message
    input_msg = input("Enter the input message in binary: ")

    # Calculate transmitted message
    crc(input_msg, poly, 1)
    transmitted_msg = input_msg + '0' * (len(poly) - 1)

    print("The transmitted message is:", transmitted_msg)

    # Input the received message
    received_msg = input("Enter the received message in binary: ")

    # Check for errors in received message
    if crc(received_msg, poly, 0):
        print("No error in data")
    else:
        print("Error in data transmission has occurred")

if __name__ == "__main__":
    main()
```

iv. Observation:

```
Enter the input message in binary: 11111
The transmitted message is: 111110000000000000000000
Enter the received message in binary: 11101
No error in data
```

```
Enter the input message in binary:
11111
The transmitted message is: 11111110001111011110
Enter the received message in binary:
11101
Error in data transmission has occurred.
```

Program 2

i. Aim of the program: Write a program for congestion control using Leaky bucket algorithm.

ii. Procedure:

~~Code : Leaky Bucket Algorithm:~~

~~def main :~~

~~Storage = 0~~

~~no. of queries = 4~~

~~bucket_size = 10~~

~~Inp-put_size = 4~~

~~out-put_size = 1~~

~~for q in range (no_of_queries):~~

~~size_left = bucket_size - Storage~~

If, $\text{PnP_pktSPc} < \text{SPc_left}$ i.e., $\text{SPc} = \text{PnP_pktSPc}$

$\text{Storage} += \text{PnP_pktSPc}$

else :

`printf("Packet loss = %PnP_pktSPc")`

`printf("Buffer SPc = %Storage} out of bucket
SPc = %bucket_SPc")`

~~$\text{Storage} -= \text{output} \times \text{PnP}$~~

~~If --name == '--main' :
main();~~

~~Output~~

Buffer size = 4 out of Bucket SPc = 10

Buffer size = 7 out of Bucket SPc = 6

1. Transfer 4 bytes. Now SPc = 6.

2. Now Buffer SPc = 10 out of Bucket SPc = 10

3. Current Packet loss = 6.4 million approx.

4. If next 20 bytes transferred then

Buffer SPc = 9 out of Bucket SPc = 10

5. Pack has completed & removed.

Now SPc = 10 - 9 = 1.0 million approx.

6. Now SPc = 1.0 million approx.

7. Now SPc = 1.0 million approx.

iii. Screen shots code:

```
storage = 0
no_of_queries = 4
bucket_size = 10
input_pkt_size = 4
output_pkt_size = 1

for i in range(no_of_queries):
    # Calculate space left in the bucket
    size_left = bucket_size - storage

    if input_pkt_size <= size_left:
        # Update storage if space is available
        storage += input_pkt_size
    else:
        # If not enough space, print packet loss
        print(f"Packet loss = {input_pkt_size}")

    # Print current buffer status
    print(f"Buffer size = {storage} out of bucket size = {bucket_size}")

    # Update storage by removing output packets
    storage -= output_pkt_size
```

iv. Observation:

```
Buffer size = 4 out of bucket size = 10
Buffer size = 7 out of bucket size = 10
Buffer size = 10 out of bucket size = 10
Packet loss = 4
Buffer size = 9 out of bucket size = 10
```

Program 3

- i. Aim of the program: Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.
- ii. Procedure:

*classmate
Date _____
Page _____*

20/12/29 LAB - 9

Apn : TCP/IP Socket Client-Server program

Client TC.Py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
ClientSocket = socket(AF_INET, SOCK_STREAM)
ClientSocket.connect((serverName, serverPort))
sentence = input('Enter file name: ')
ClientSocket.send(sentence.encode())
print('From server: ', ClientSocket.recv(1024).decode())
ClientSocket.close()
```

Server TC.Py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
ServerSocket = socket(AF_INET, SOCK_STREAM)
ServerSocket.bind((serverName, serverPort))
ServerSocket.listen(1)

while 1:
    print('Server is ready to receive')
    connectionSocket, addr = ServerSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    connectionSocket.close()
```

iii. Screen shot code:

```
from socket import *

serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)

while 1:
    print("The server is ready to receive")

    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()

    file = open(sentence, "r")
    l = file.read(1024)
    connectionSocket.send(l.encode())

    print(f'\nSent contents of {sentence}')

    file.close()
    connectionSocket.close()
```

```
from socket import *
serverName = '127.0.0.1' # IP address of the server (localhost)
serverPort = 12000         # Port number on which the server is listening
clientSocket = socket(AF_INET, SOCK_STREAM) # Create a TCP socket (IPv4, TCP)

clientSocket.connect((serverName, serverPort)) # Connect to the server

# Input the file name from the user
sentence = input("\nEnter file name: ")

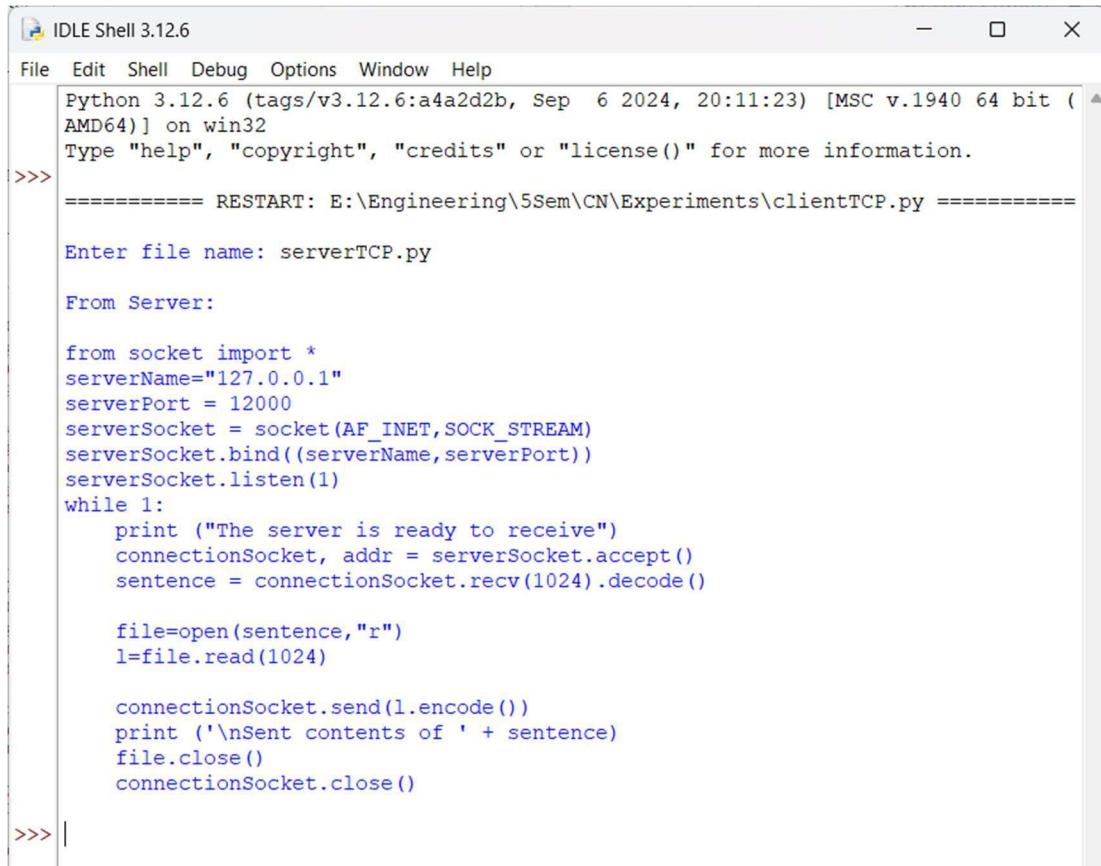
# Send the file name to the server
clientSocket.send(sentence.encode())

# Receive the file contents from the server
filecontents = clientSocket.recv(1024).decode()

# Print the contents received from the server
print('\nFrom Server:\n')
print(filecontents)

# Close the socket
clientSocket.close()
```

iv. Observation:

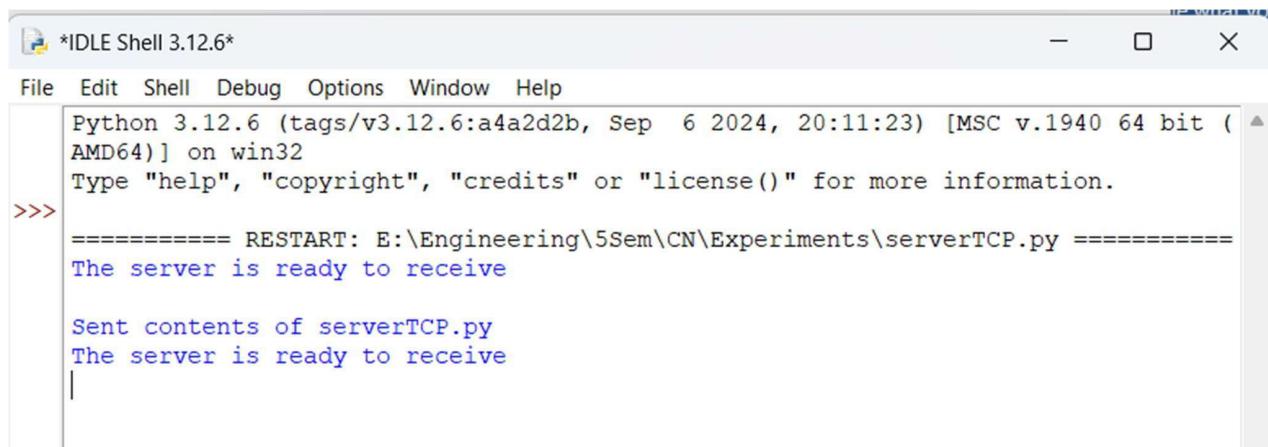


IDLE Shell 3.12.6

```
File Edit Shell Debug Options Window Help
Python 3.12.6 (tags/v3.12.6:a4a2d2b, Sep  6 2024, 20:11:23) [MSC v.1940 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> ===== RESTART: E:\Engineering\5Sem\CN\Experiments\clientTCP.py =====
Enter file name: serverTCP.py
From Server:
from socket import *
serverName="127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print ("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()

    file=open(sentence,"r")
    l=file.read(1024)

    connectionSocket.send(l.encode())
    print ('\nSent contents of ' + sentence)
    file.close()
    connectionSocket.close()
>>>
```



IDLE Shell 3.12.6

```
File Edit Shell Debug Options Window Help
Python 3.12.6 (tags/v3.12.6:a4a2d2b, Sep  6 2024, 20:11:23) [MSC v.1940 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> ===== RESTART: E:\Engineering\5Sem\CN\Experiments\serverTCP.py =====
The server is ready to receive
Sent contents of serverTCP.py
The server is ready to receive
```

Program 4

- i. Aim of the program:

Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

- ii. Procedure:

10/11/24
CAB - 7
Aim: UDP Sockets

classmate
Data
Page

clientUDP.py

```
from socket import *
ServerName = '127.0.0.1'
ServerPort = 12000
```

Client Socket. send to Client (Sentence)
(UTF-8)

```
ClientSocket. close()
Client. Socket. close()
```

serverUDP.py.

```
from socket import *
ServerPort = 12000
ServerPort = '127.0.0.1'
ServerSocket = socket.bind ('127.0.0.1', ServerPort)

while 1:
    Sentence, ClientAddress = ServerSocket. recvfrom(2048)
    com = file. read(2048)

    ServerSocket. sendto((key). encode('utf-8'), ClientAddress)
    file. close()
```

iii. Screen shots code:

```
clientUDP.py
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)

sentence = input("\nEnter file name: ")

clientSocket.sendto(bytes(sentence, "utf-8"), (serverName, serverPort))

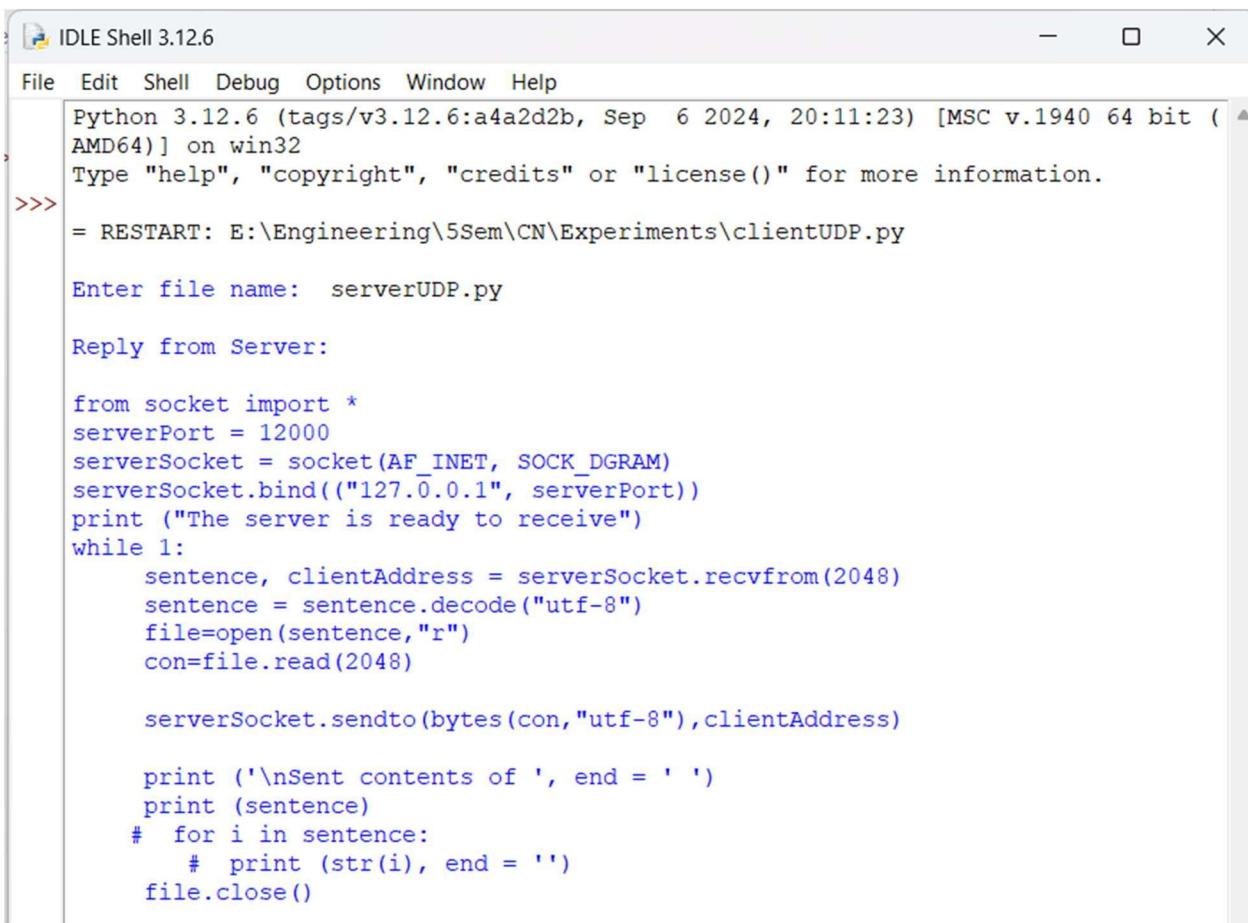
filecontents, serverAddress = clientSocket.recvfrom(2048)
print ("\nReply from Server:\n")
print (filecontents.decode("utf-8"))
# for i in filecontents:
#     # print(str(i), end = "")
clientSocket.close()
clientSocket.close()

serverUDP.py
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence,"r")
    con=file.read(2048)

    serverSocket.sendto(bytes(con, "utf-8"), clientAddress)

    print ("\nSent contents of', end = ' ')
    print (sentence)
    # for i in sentence:
    #     # print (str(i), end = "")
    file.close()
```

iv. Observation:



```
IDLE Shell 3.12.6
File Edit Shell Debug Options Window Help
Python 3.12.6 (tags/v3.12.6:a4a2d2b, Sep 6 2024, 20:11:23) [MSC v.1940 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\Engineering\5Sem\CN\Experiments\clientUDP.py

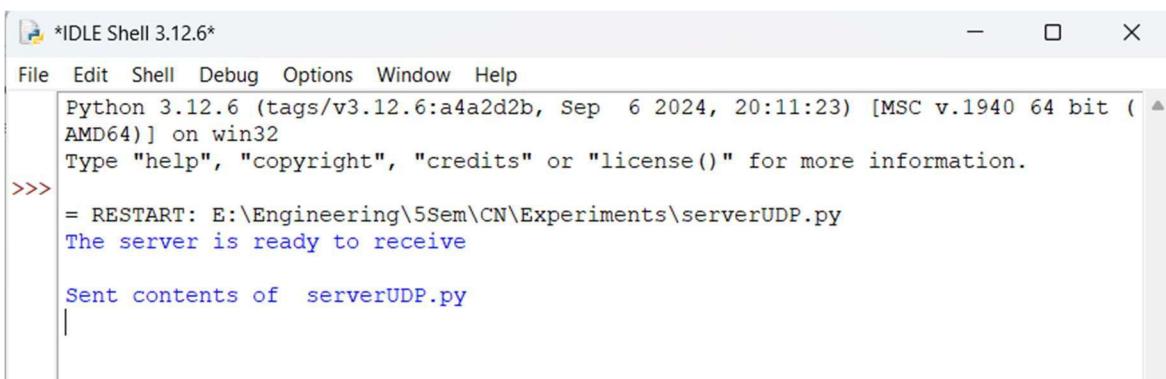
Enter file name: serverUDP.py

Reply from Server:

from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file=open(sentence, "r")
    con=file.read(2048)

    serverSocket.sendto(bytes(con,"utf-8"),clientAddress)

    print ('\nSent contents of ', end = ' ')
    print (sentence)
#   for i in sentence:
#       #   print (str(i), end = '')
    file.close()
```



```
*IDLE Shell 3.12.6*
File Edit Shell Debug Options Window Help
Python 3.12.6 (tags/v3.12.6:a4a2d2b, Sep 6 2024, 20:11:23) [MSC v.1940 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: E:\Engineering\5Sem\CN\Experiments\serverUDP.py
The server is ready to receive

Sent contents of serverUDP.py
```