VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

COMPUTER NETWORKS

Submitted by

HEMA P(1BM22CS111)

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 Sep
2024-Jan 2025

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 NETWORKS**" carried out by Hema P (1BM22CS111) who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Computer Networks Lab - (23CS5PCCON)** work prescribed for the said degree.

Spoorthi D M

Assistant Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Kavitha Sooda

Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

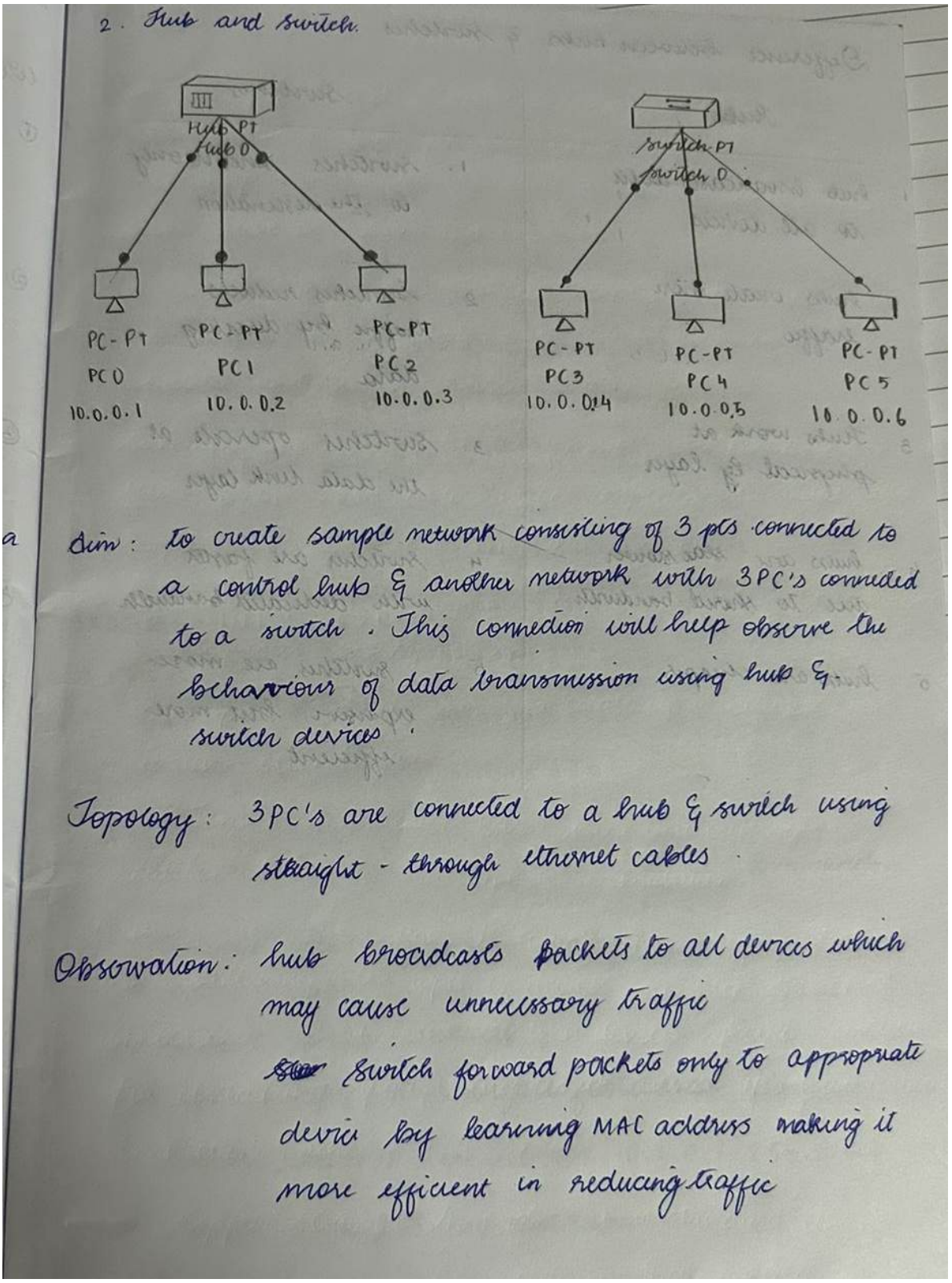
CYCLE 1

INDEX

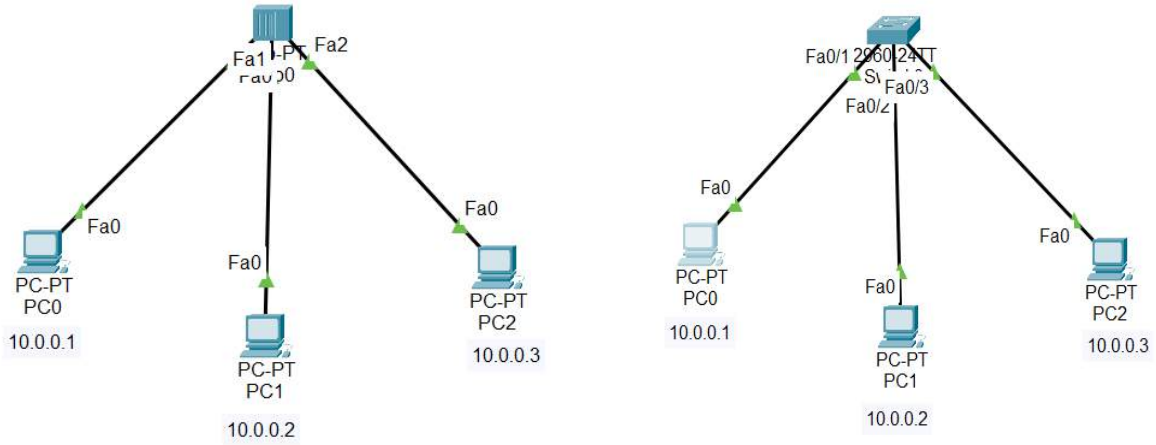
CYCLE 2

Cycle -1
Experiment 1:
Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message

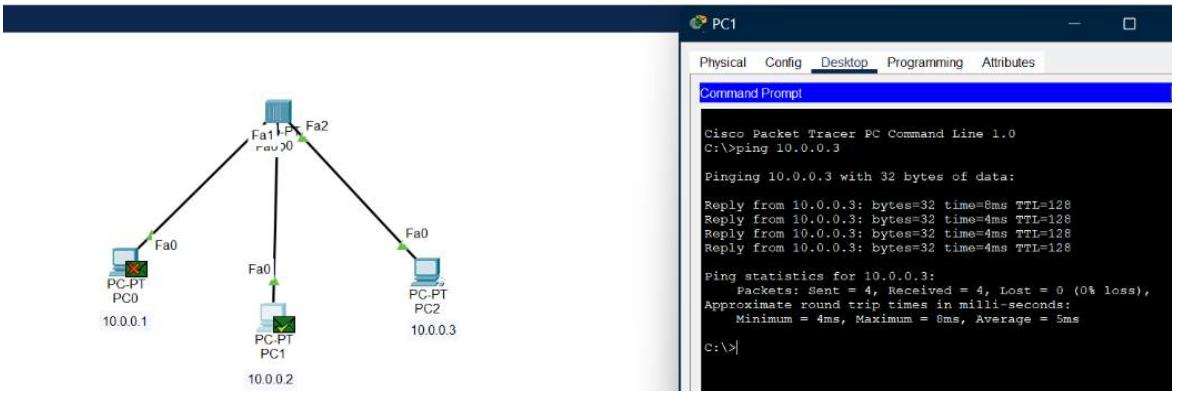
Observation:

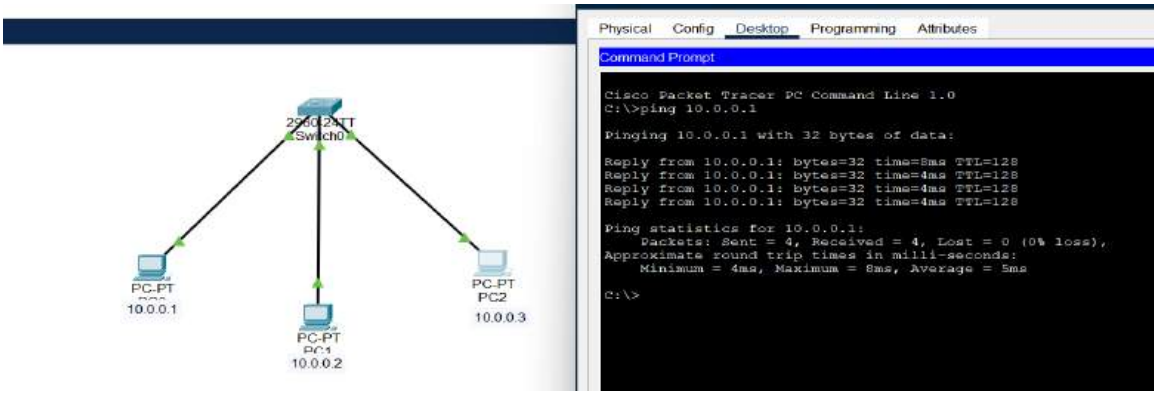


Topology:



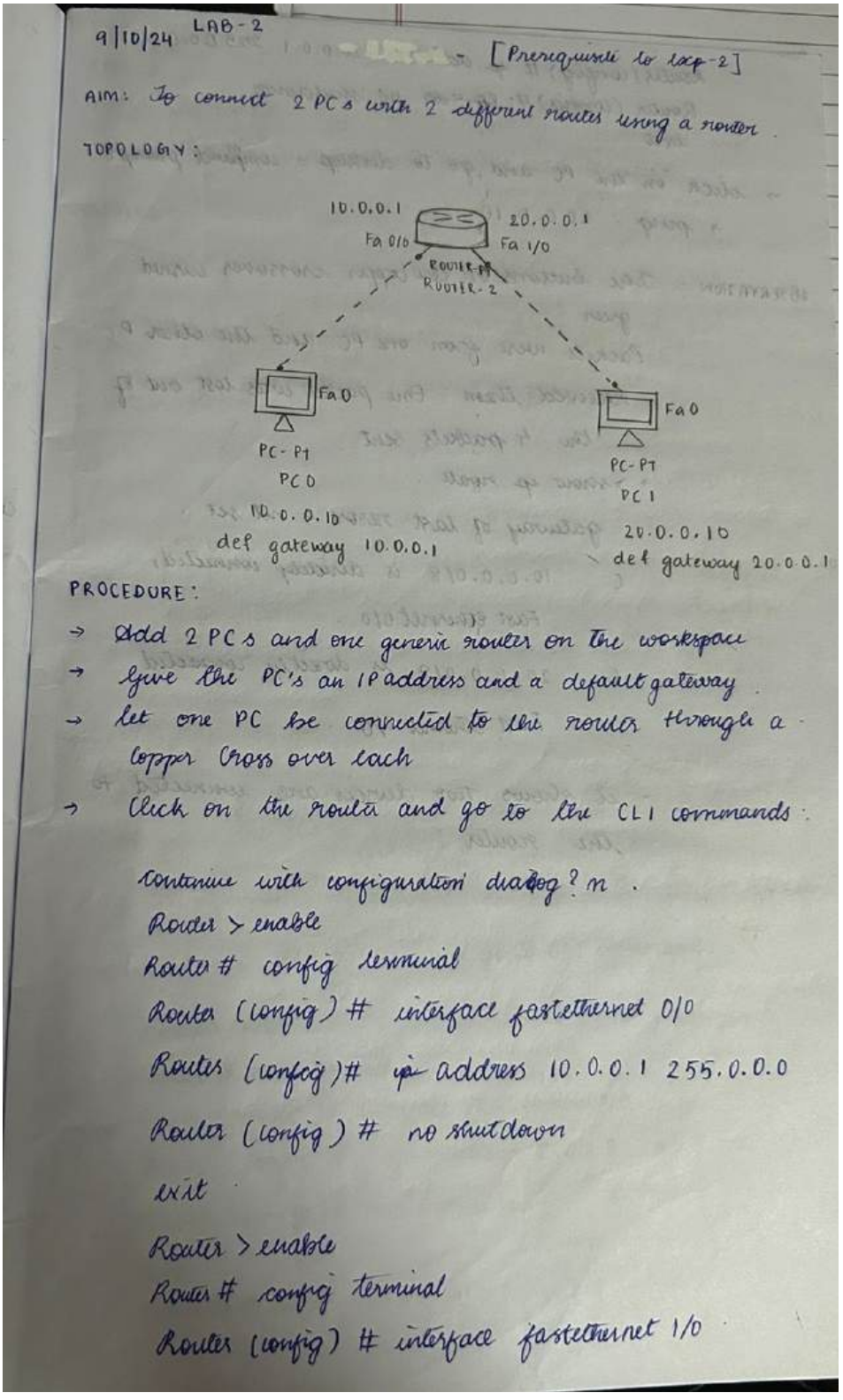
Output:





Experiment 2:
Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply

Observation Book:



9/10/24 LAB-2 - [Prerequisite to exp-2]

AIM: To connect 2 PCs with 2 different routers using a router.

TOPOLOGY:

PROCEDURE:

→ Add 2 PCs and one generic router on the workspace
→ Give the PC's an IP address and a default gateway.
→ Let one PC be connected to the router through a copper Cross over each
→ Click on the route and go to the CLI commands:

Continue with configuration dialog? n.
Router > enable
Router # config terminal
Router (config) # interface fastethernet 0/0
Router (config)# ip address 10.0.0.1 255.0.0.0
Router (config) # no shutdown
exit
Router > enable
Router # config terminal
Router (config) # interface fastethernet 1/0

Router (config) # ip address 20.0.0.1 255.0.0.0

Router (config) # no shutdown

exit.

→ click on the PC and go to desktop → command prompt.

→ ping 20.0.0.10

OBSERVATION : The buttons on the copper crossover turned green.

Packets were from one PC and the other PC received them. One packet was lost out of the 4 packets sent.

→ >show ip route.

gateway of last resort is not set.

C 10.0.0.0/8 is directly connected, Fast Ethernet 0/0.

C 20.0.0.0/8 is directly connected Fast Ethernet 1/0.

- it shows two devices are connected to the router.

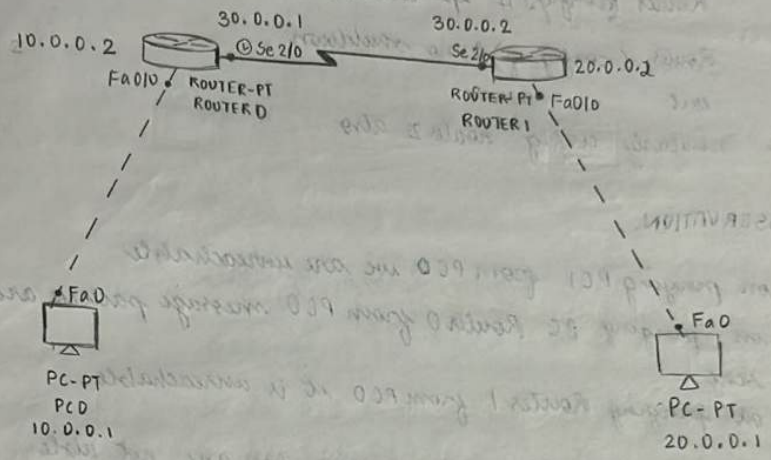
16/10/24

LAB-3

- EXP - 2 -

AIM: To connect two Routers serially and connect end devices to respective routers.

TOPOLOGY



PROCEDURE

→ Add two generic PCs and two generic routers in the workspace

→ Give the PCs a default gateway and IP address

→ each PC is connected with a copper wire to the respective router.

→ click on the router and go to CLI commands.

```
Router > enable
Router # config terminal
Router (config)# interface fastethernet 0/0
Router (config)# ip address 10.0.0.2 255.0.0.0
Router (config)# no shutdown
exit
```

→ Similarly connect PC 2 also

→ for router to router

- Router > enable
  Router # config terminal
  Router (config) # interface serial 2/0
  Router (config) # ip address 30.0.0.1 255.0.0.0
  Router (config-if) # no shutdown
  exit

→ similarly config. router 2 also.

OBSERVATION:

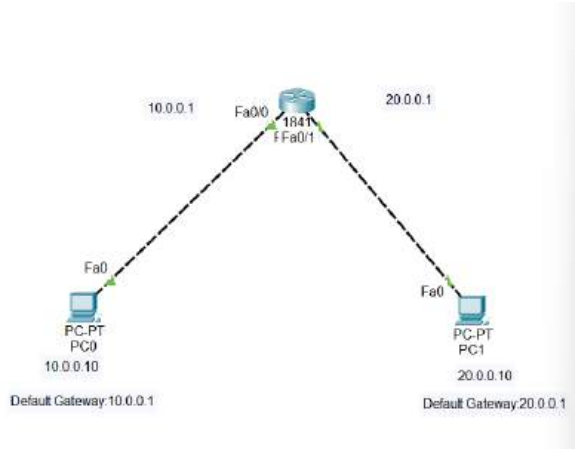
→ on pinging PC1 from PC0 we are unreachable

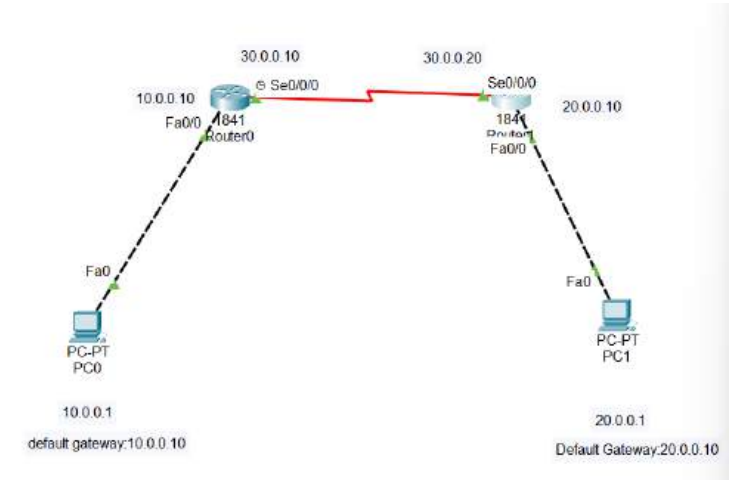
→ on pinging Router0 from PC0 message packets are sent

→ on pinging Router1 from PC0 it is unreachable.

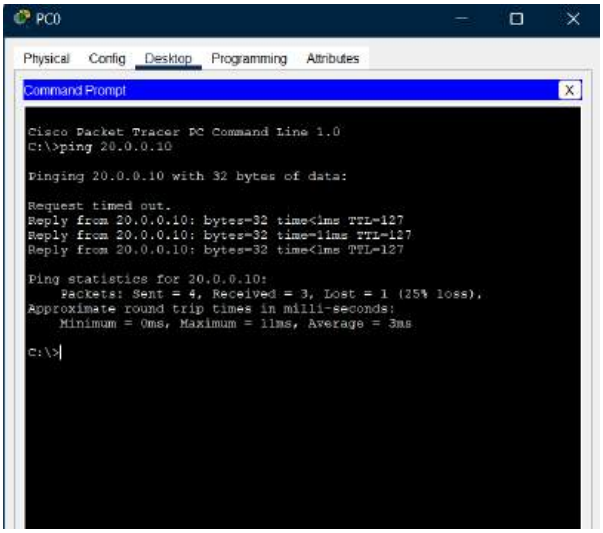
Therefore from the end devices we are not able to send or receive message packets through the routers.

Topology:

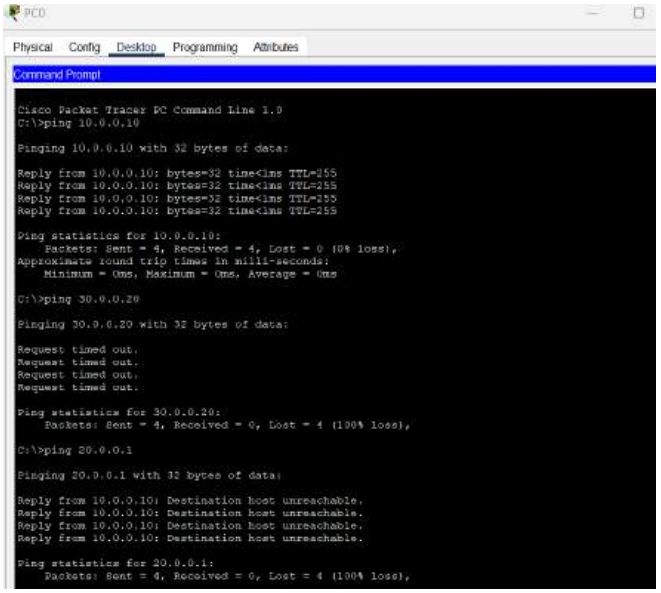




Output:



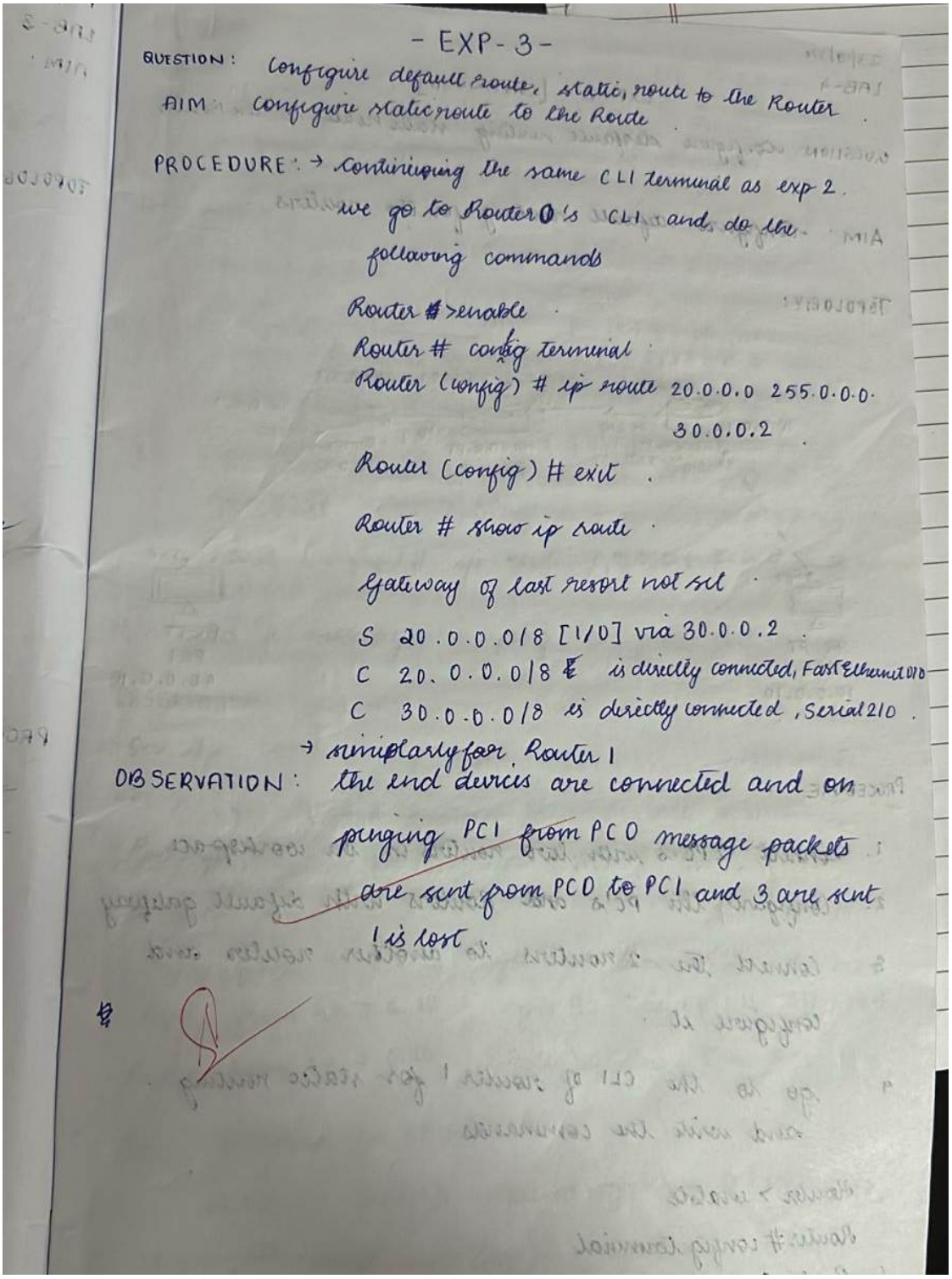

Router0
Physical
Config
CLI
Attributes
IOS Command Line Interface
Press RETURN to get started.
Router>enable
Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
N1 - OSPF NSSA external type 1, N2 - 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
C 30.0.0.0/8 is directly connected, Serial0/0/0


PC0
Physical
Config
Desktop
Programming
Attributes
Command Prompt
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.10
Pinging 10.0.0.10 with 32 bytes of data:
Reply from 10.0.0.10: bytes=32 time<1ms TTL=255
Reply from 10.0.0.10: bytes=32 time<1ms TTL=255
Reply from 10.0.0.10: bytes=32 time<1ms TTL=255
Reply from 10.0.0.10: bytes=32 time<1ms TTL=255
Ping statistics for 10.0.0.10:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
C:\>ping 30.0.0.20
Pinging 30.0.0.20 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 30.0.0.20:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\>ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Ping statistics for 20.0.0.1:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

Experiment 3:

Configure default route, static route to the Router

Observation Book:



- EXP-3 -

QUESTION: Configure default route, static route to the Router.

AIM: Configure static route to the Route.

PROCEDURE: → Continuing the same CLI terminal as exp 2.
we go to Router0's CLI and do the following commands

Router #>enable.
Router # config terminal.
Router (config) # ip route 20.0.0.0 255.0.0.0 30.0.0.2.
Router (config) # exit.
Router # show ip route.
gateway of last resort not set.
S 20.0.0.0/8 [1/0] via 30.0.0.2.
C 20.0.0.0/8 is directly connected, FastEthernet0/0
C 30.0.0.0/8 is directly connected, Serial2/0.

→ similarly for Router 1

OBSERVATION: the end devices are connected and on pinging PC1 from PC0 message packets are sent from PC0 to PC1 and 3 are sent 1 is lost.

23/10/24

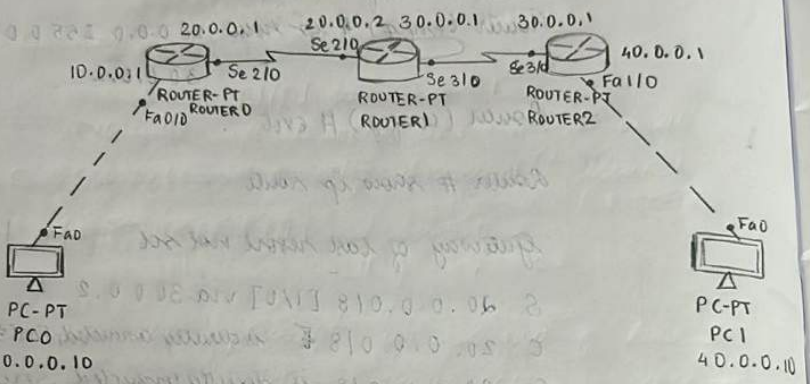
LAB-4

-EXP-3- [CONTINUED]

QUESTION: Configure default routing, static route to the router

AIM: Configure default routing for the routers.

TOPOLOGY:



PROCEDURE

1. Connect 2 PC's with two routers in the workspace
2. Configure the PC's and Routers with default gateway
3. Connect the 2 routers to another router and configure it.
4. go to the CLI of router 1 for static routing. and write the commands

```
Router > enable
Router # config terminal
Router (config) # ip route 10.0.0.0 255.0.0.0 20.0.0.1
Router (config) # exit
```

Router# config terminal

Router (config)# ip route 40.0.0.0 255.0.0.0 30.0.0.2

Router (config)# exit.

5. go to the CLI of Router 0 for default routing and the following commands.

Roula > enable

Router # config terminal

Router (config)# ip route 0.0.0.0 0.0.0.0 20.0.0.2

Router (config)# exit.

6. similarly do for router 1 while keeping the ip route command as.

Router (config)# ip route 0.0.0.0 0.0.0.0 30.0.0.1

7. Send a message i.e ping PC1 from PC0.

OBSERVATION:

⇒ On the command prompt:

PC > ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data

Request timed out.

Reply from 40.0.0.10: bytes = 32 time = 12 ms TTL = 125

Reply from 40.0.0.10: bytes 32 time = 12 ms TTL = 125

Reply from 40.0.0.10: bytes 32 time = 9 ms TTL = 125

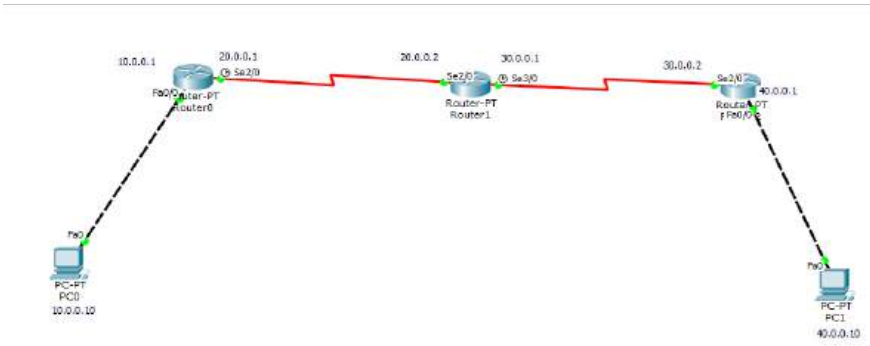
Ping statistics for 40.0.0.10:

Packets: Sent = 4, Received = 3, lost = 1 (25% loss),

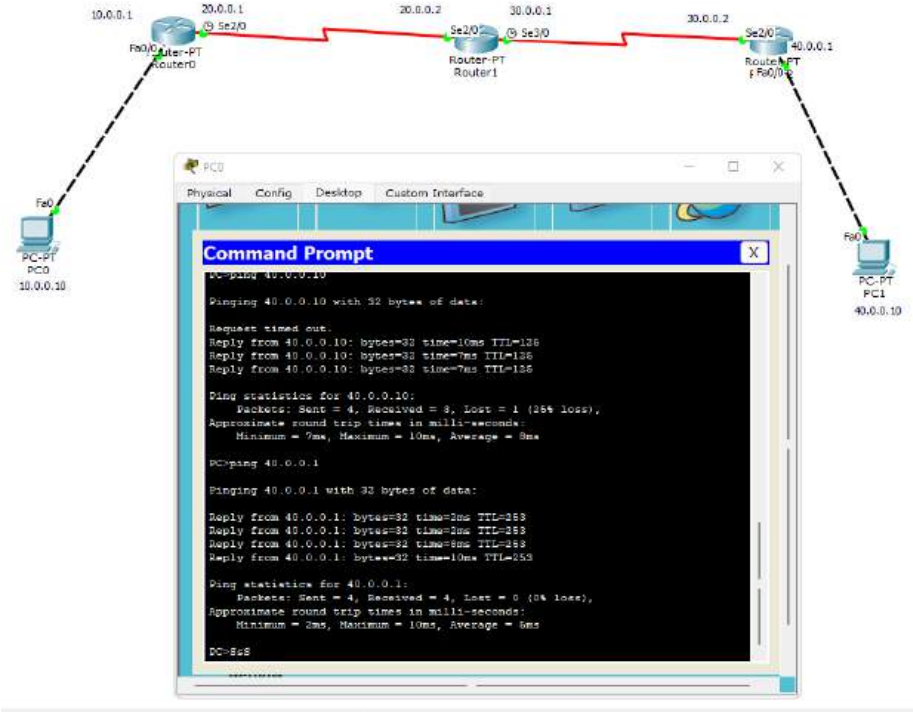
Approximate round trip times in milliseconds:

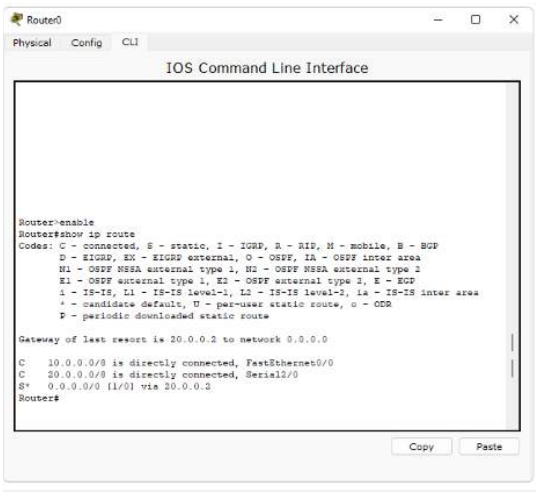
Minimum = 2 ms, maximum = 12 ms, Average = 7 ms.

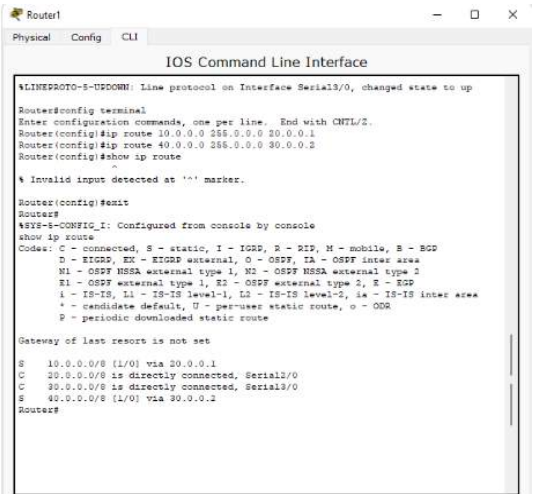
Topology:



Output:

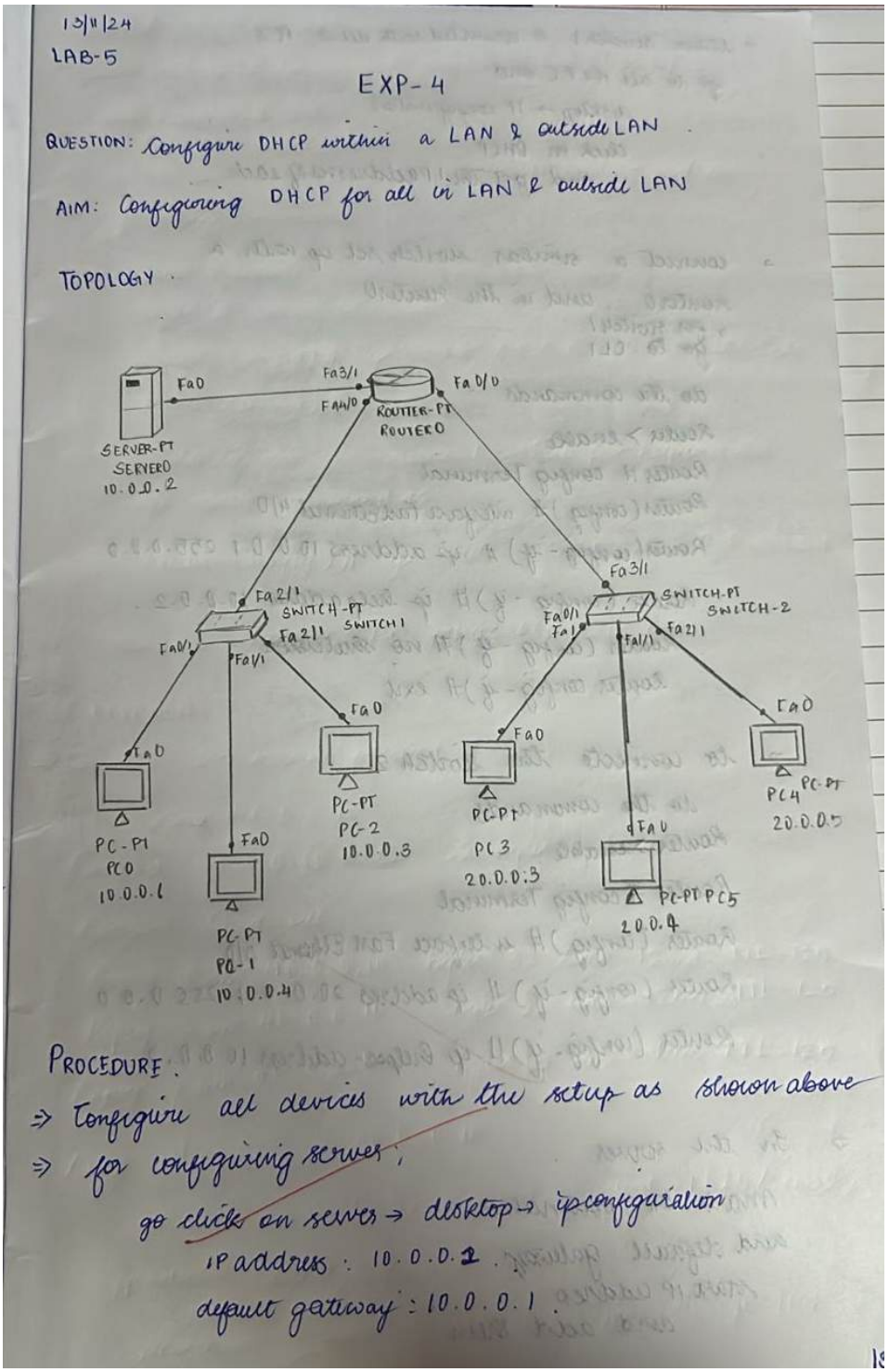






Experiment 4:
Configure DHCP within a LAN and outside LAN.

Observation Book:



→ When switch 1 is connected with all 3 PC's
go to all the PC and
desktop → IP configuration
click on DHCP
and get the IP addresses of each

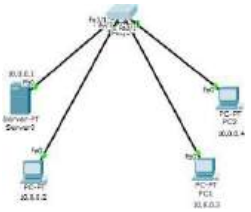
→ connect a similar switch set up with a router0 and in the router0
→ for switch1
go to CLI
do the commands
Router > enable
Router # config Terminal
Router (config) # interface FastEthernet 4/0
Router (config-if) # ip address 10.0.0.1 255.0.0.0
Router (config-if) # ip help address 10.0.0.2
Router (config-if) # no shutdown
Router config-if) # exit

→ to connects the switch 2
do the commands
Router > enable
Router # config Terminal
Router (config) # interface FastEthernet 0/0
Router (config-if) # ip addres 20.0.0.1 255.0.0.0
Router (config-if) # ip helper-address 10.0.0.2

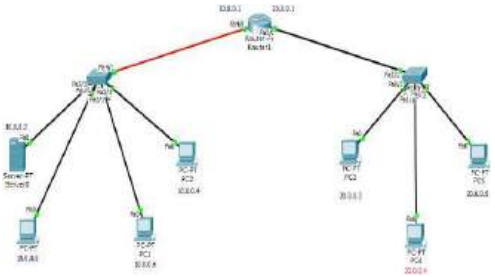
⇒ In the server
Mark the pool names
and default gateways ;
start IP address
and add them

Topology:

(within Lan)

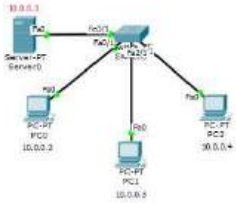


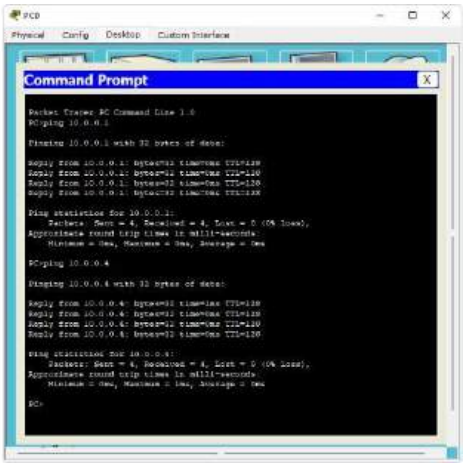
(outside Lan)



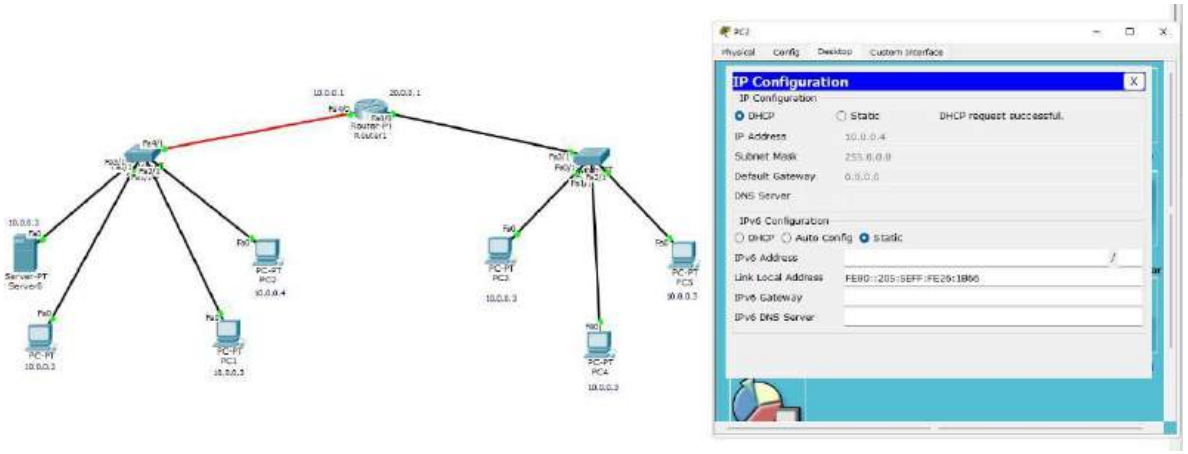
Output:

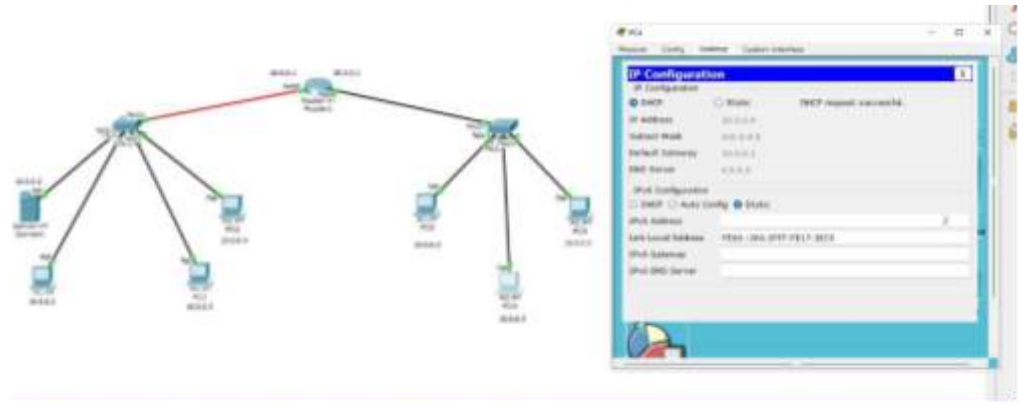
(within Lan)

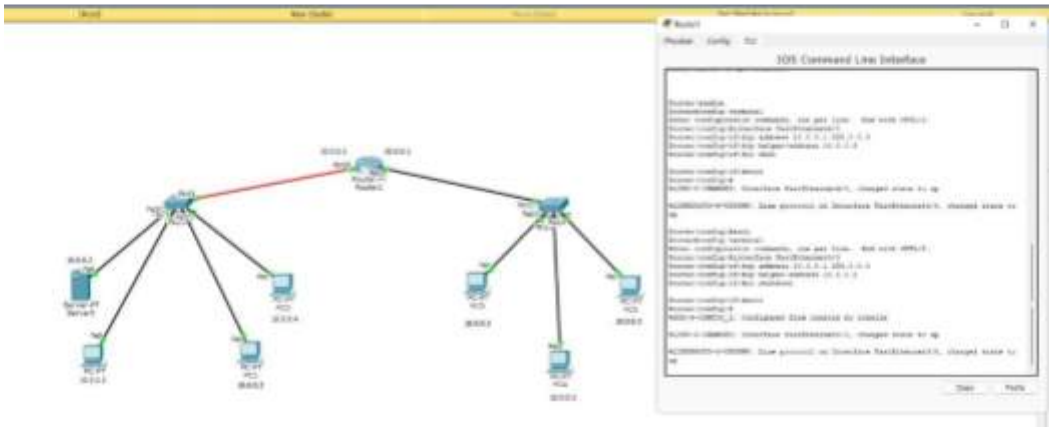




(outside Lan)

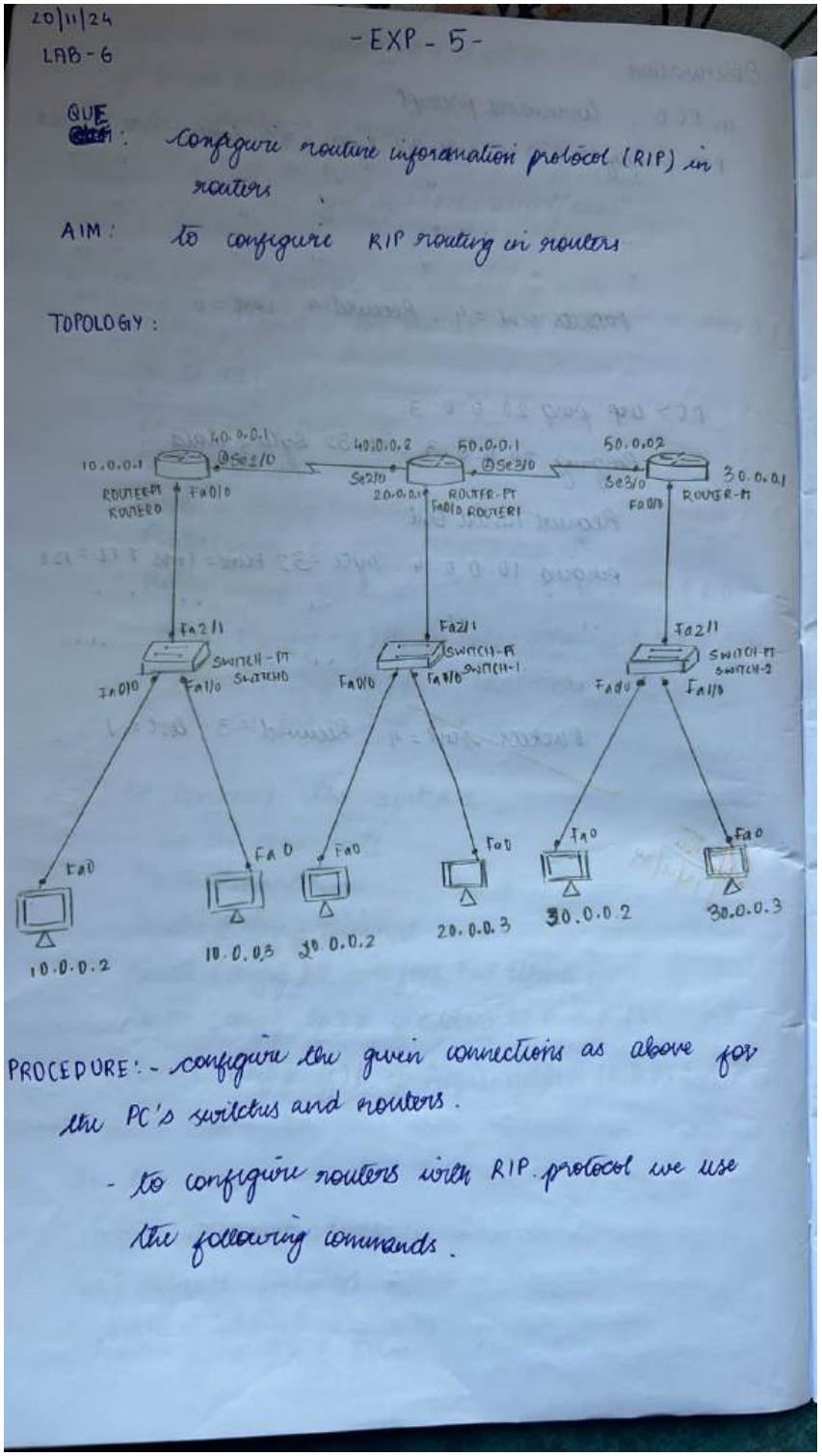

IP Configuration
DHCP
Static
DHCP request successful.
IP Address
Subnet Mask
Default Gateway
DNS Server
IPv6 Configuration
IPv6 Address
Link Local Address
IPv6 Gateway
IPv6 DNS Server


IP Configuration


IOS Command Line Interface

Experiment 5:
Configure RIP routing Protocol in Routers

Observation Book:



in Router 0's CLI

Router> enable

Router# config terminal

Router (config)# router rip

Router(config-router)# network 10.0.0.0

Router (config-router)# network 40.0.0.0

Router (config-router)# exit

in Router 1's CLI

Router (config)# router rip

Router (config-router)# network 40.0.0.0

Router (config-router)# network 50.0.0.0

Router (config-router)# network 20.0.0.0

Router (config-router)# exit

in Router 2's CLI

Router (config)# router rip

Router (config-router)# network 50.0.0.0

Router (config-router)# network 30.0.0.0

Router (config-router)# exit

OUTPUT:

in command prompt of PC0

ping > 30.0.0.3

Request Time out

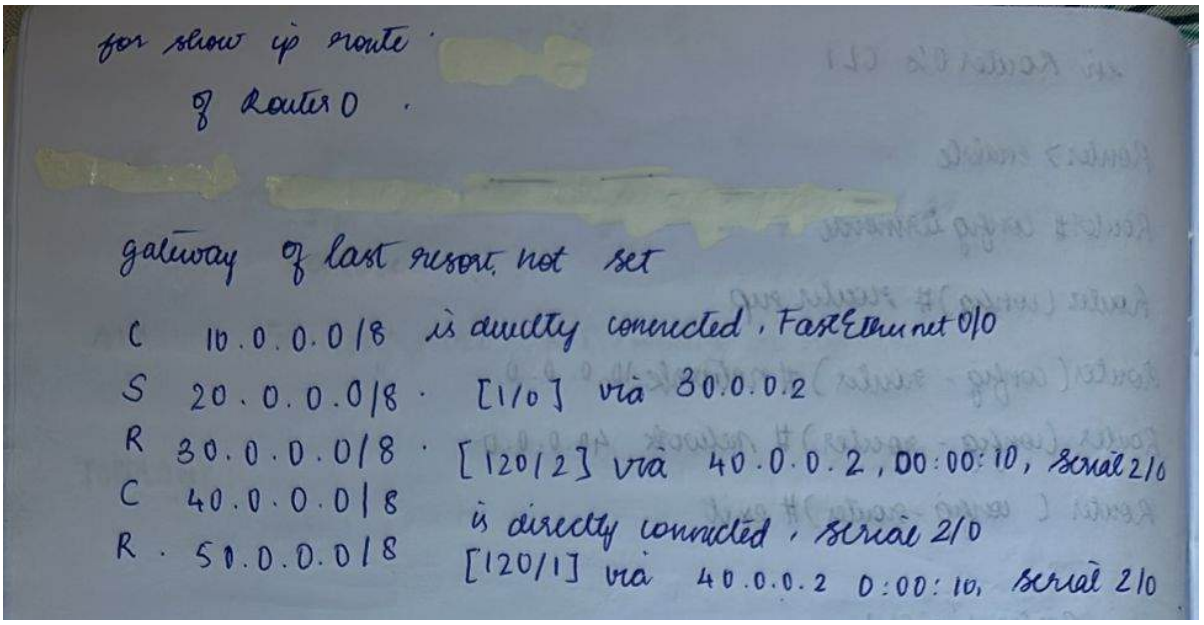
Reply from 30.0.0.3 bytes = 32 time = 12 ms TTL = 125

"

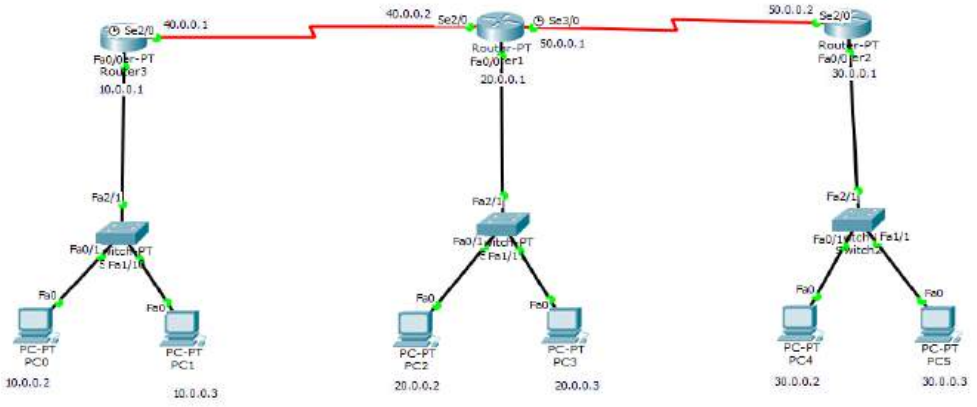
"

"

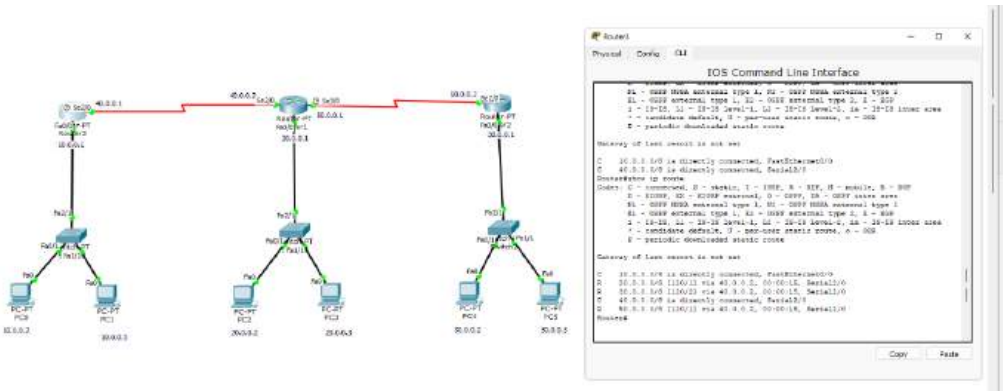
Packets Send = 4, Received = 4, Lost = 0 (0% loss)

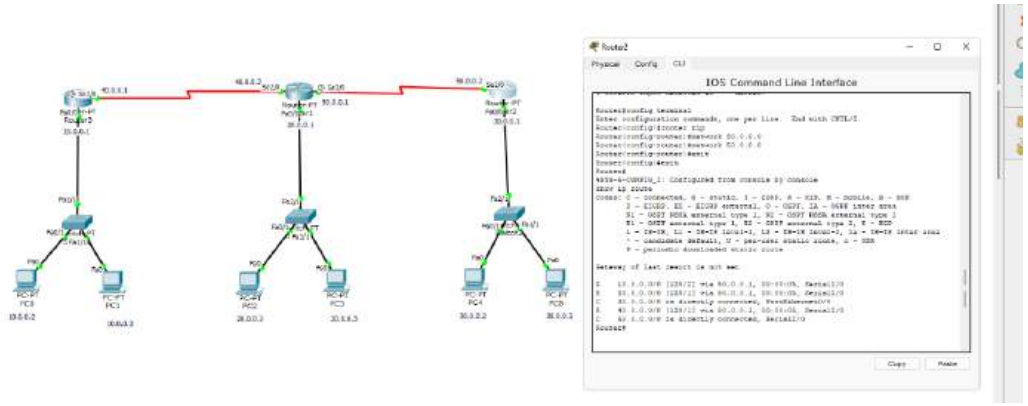


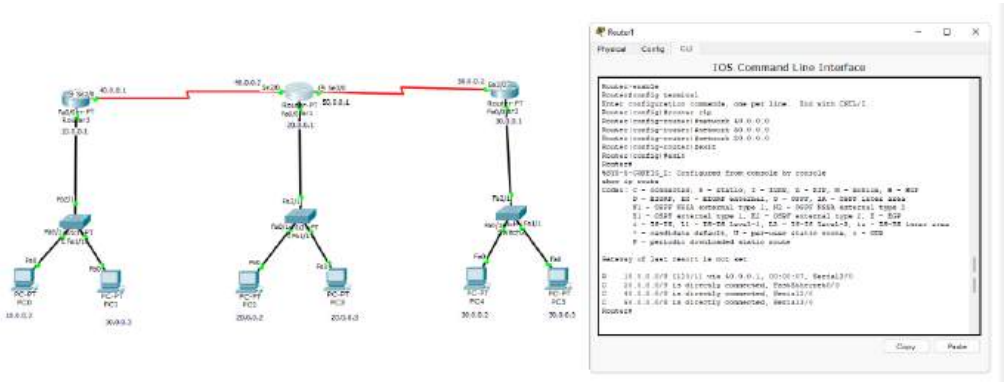
Topology:

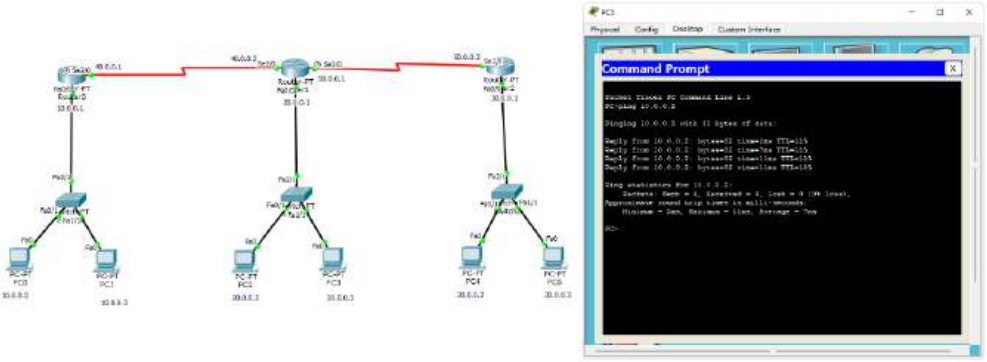


Output:



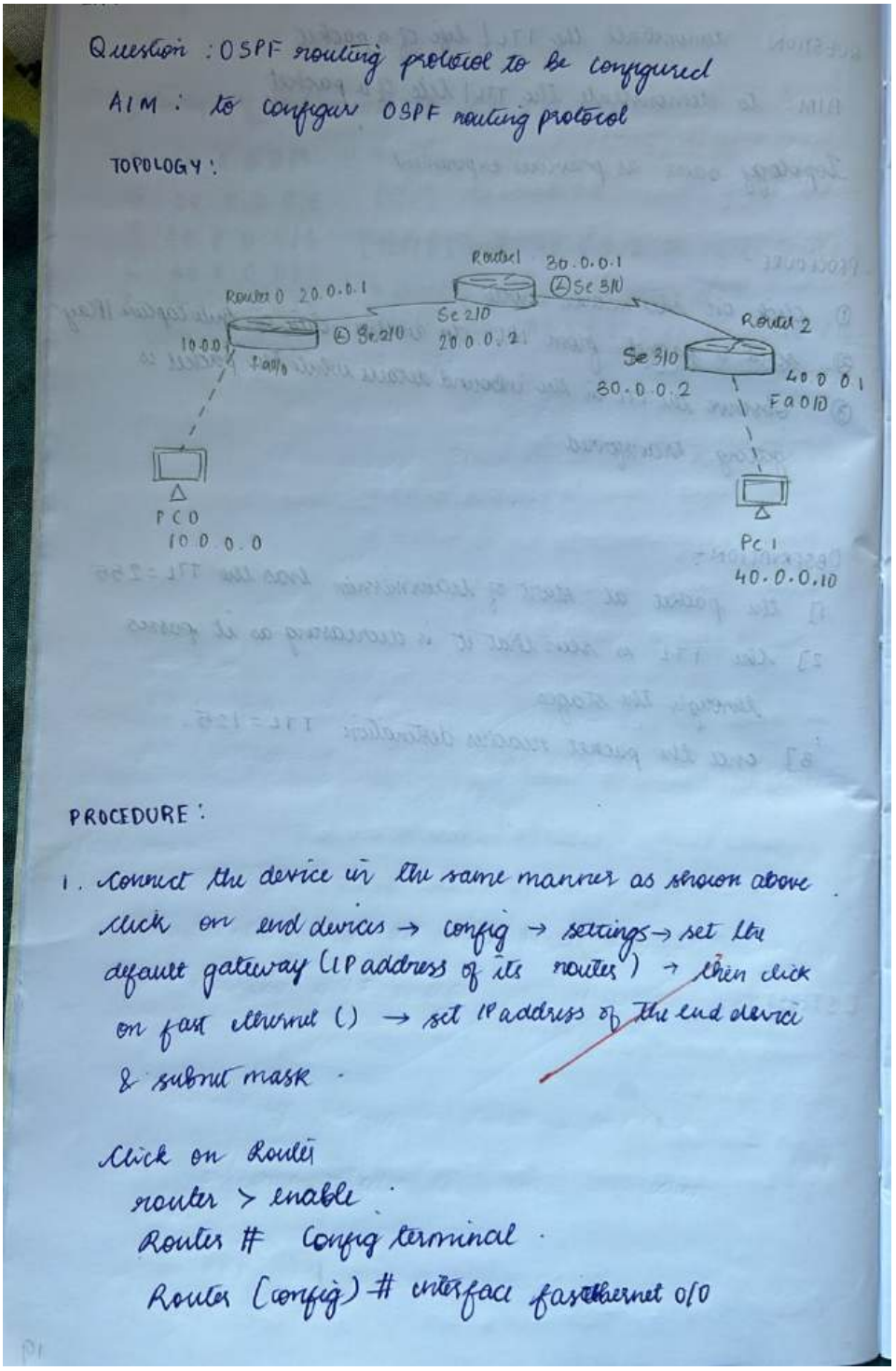

Router2
Physical
Config
CLI
IOS Command Line Interface
Copy
Paste


Router1
Physical
Config
CLI
IOS Command Line Interface
Copy
Paste


PC3
Physical
Config
Desktop
Custom Interface
Command Prompt

Experiment 6:
Configure OSPF routing protocol

Observation Book:



Router (config-if) # ip address 10.0.0.1 255.0.0.0.

router (config) # no shut

exit.

Similarly configure Router 2 to PC1 & configure router to router connections

2. for router1

R(config)# interface serial 2/0

R(config-if)# ip address 20.0.0.2 255.0.0.0

R(config-if) # encapsulation ppp

R(config-if) # no shut

R(config-if) # exit

~~serial for router~~.

R(config)# interface 3/0

R(config-if) # ip address 30.0.0.1 255.0.0.0

encapsulation ppp.

clock rate 64000

no shut

exit

do similiarly for router 0 & Router 2

3. in router 0

R(config)# router ospf 1

R(config-router) # router-id 1.1.1.1

" # network 10.0.0.0 0.255.255.255 area 3

" # " 20.0.0.0 " area 1

" # exit

In router 1

R (config) # router ospf 1

router-id 2.2.2.2

network 20.0.0.0 0.255.255.255 area 1

" 80.0.0.0 " area 0

exit

in router 2

R (config) # router ospf 1

router-id 3.3.3.3

network 30.0.0.0 0.255.255.255 area 0

" 40.0.0.0 "

exit

4.

RD (config-if) # interface loopback 0

RD (config-if) # ip add 172.16.1.252 255.255.0.0

no shut

Similarly

R1 (config) # ipadd 172.16.1.253 255.255.0.0

R2 (config-if) # ipadd 172.16.1.254 255.255.0.0

Similarly configure Router 2 to PC1 &

Configure routes to router connection

Step

5. Create virtual link btw R1 & R0

in R0

R0(config)# router ospf 1.

R0(config router)# area 1 virtual link 2.2.2.2

in R1

R1(config-router)# area 1 virtual link

exit.

-OBSERVATION:

in R2.

Router# show ip route

O IA 20.0.0.0/8 [110/128] via 30.0.0.1 09:57:23

C 40.0.0.0/8 is directly connected, FastEthernet 0/0 serial 2/0

O IA 10.0.0.0/8 [110/128] via 30.0.0.1 00:57:05, serial 2/0

C 30.0.0.0/8 is directly connected, serial 2/0.

C 172.16.0.0/16. is directly connected, Loopback 0

similarly the output is shown for Router 0 & 1

Ping output:

(from PC0 to PC1)

PC0 → command prompt.

C:\> ping 40.0.0.10

pinging 40.0.0.10 with 32 bytes of data

Request timed out.

Reply from 40.0.0.10 bytes = 32 time = 21ms TTL = 125

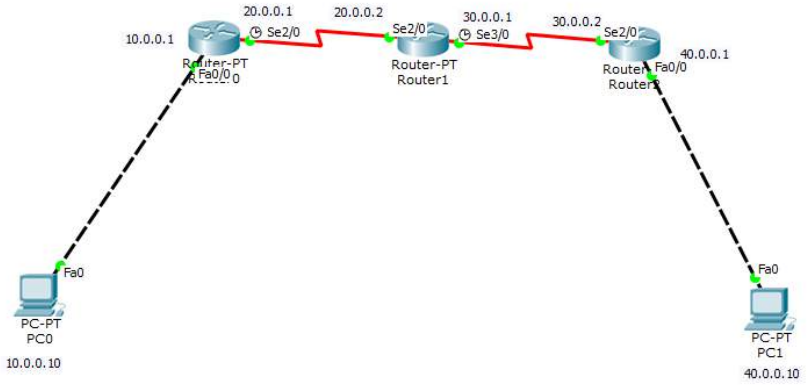
Reply from 40.0.0.10 " " " "

" " " " " " " "

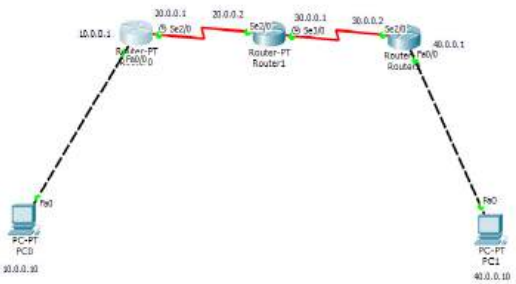
Packets sent = 4, Received = 3 lost = 1 (25% loss)

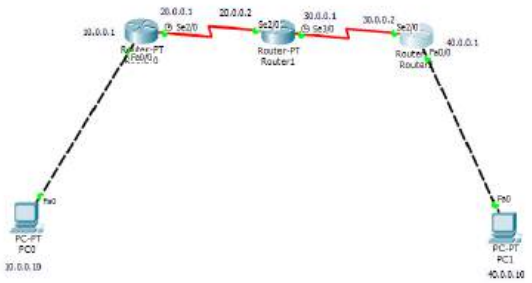
18/12/24

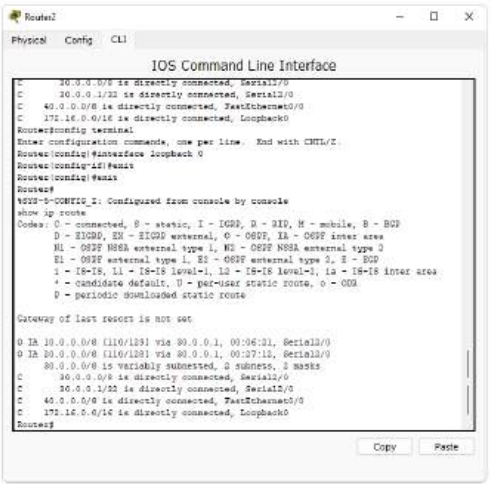
Topology:

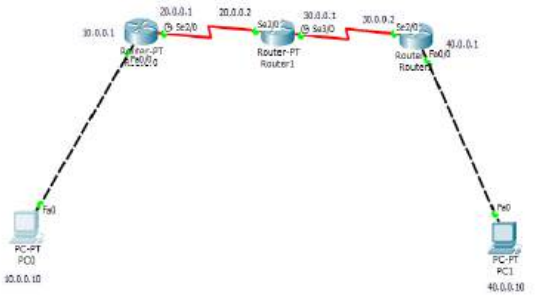


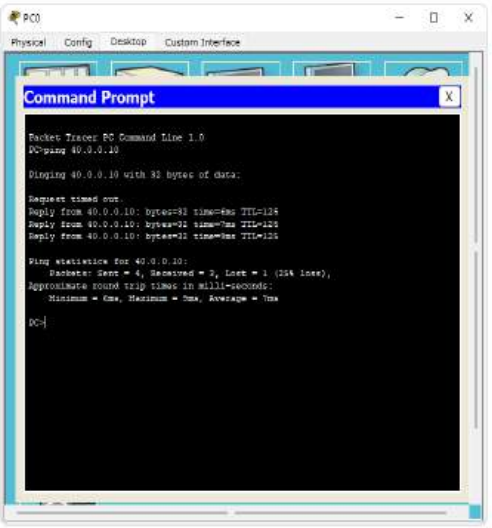
Output:



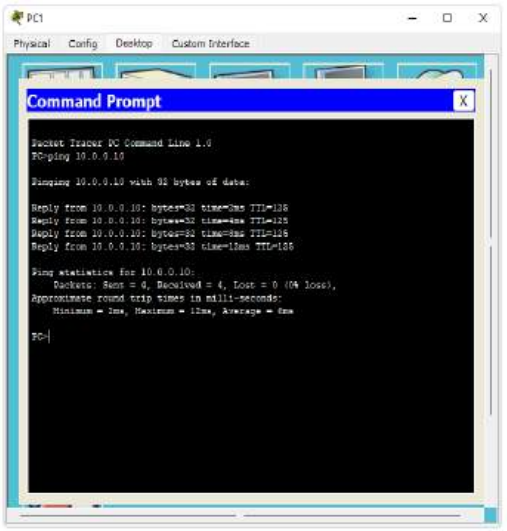





20.0.0.1
20.0.0.2
30.0.0.1
30.0.0.2
10.0.0.1
Se2/0
Se3/0
Router-PT
Router1
Router-PT
Router2
40.0.0.1
Fa0/0
Fa0
PC-PT
PC0
10.0.0.10
PC-PT
PC1
40.0.0.10


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.10
Pinging 40.0.0.10 with 32 bytes of data:
Request timed out.
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=7ms TTL=125
Reply from 40.0.0.10: bytes=32 time=8ms TTL=125
Ping statistics for 40.0.0.10:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 6ms, Maximum = 8ms, Average = 7ms
PC>

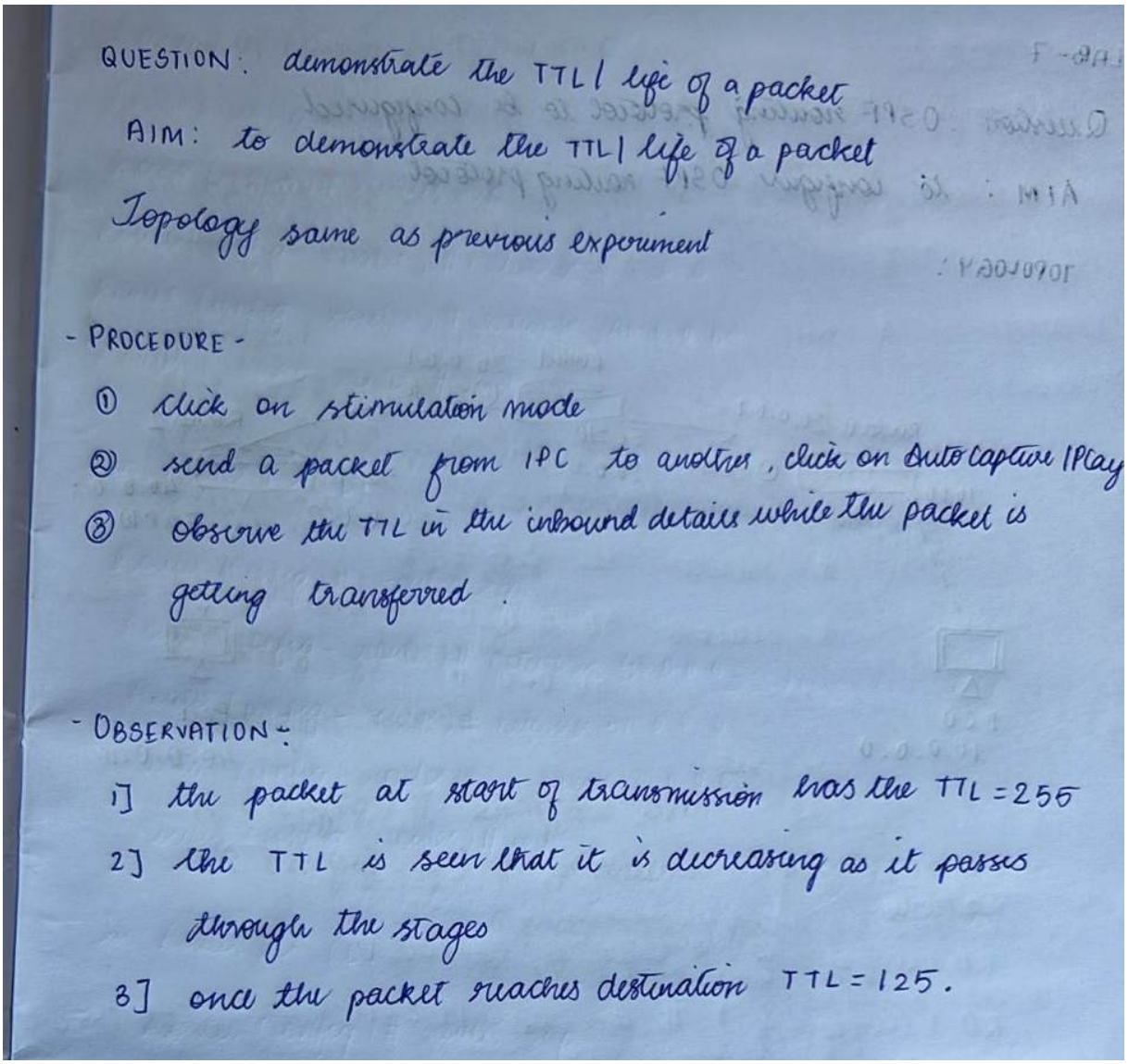

20.0.0.1
20.0.0.2
30.0.0.1
30.0.0.2
10.0.0.1
Se2/0
Se3/0
Router-PT
Router1
Router-PT
Router2
40.0.0.1
Fa0/0
Fa0
PC-PT
PC0
10.0.0.10
PC-PT
PC1
40.0.0.10


PC1
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.10
Pinging 10.0.0.10 with 32 bytes of data:
Reply from 10.0.0.10: bytes=32 time=2ms TTL=126
Reply from 10.0.0.10: bytes=32 time=4ms TTL=125
Reply from 10.0.0.10: bytes=32 time=9ms TTL=126
Reply from 10.0.0.10: bytes=32 time=12ms TTL=126
Ping statistics for 10.0.0.10:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 2ms, Maximum = 12ms, Average = 6ms
PC>

Experiment 7:

Demonstrate the TTL/ Life of a Packet

Observation Book:



QUESTION: demonstrate the TTL / life of a packet

AIM: to demonstrate the TTL / life of a packet

Topology same as previous experiment

- PROCEDURE -

① click on stimulation mode

② send a packet from PC to another, click on auto capture / play

③ observe the TTL in the inbound details while the packet is getting transferred.

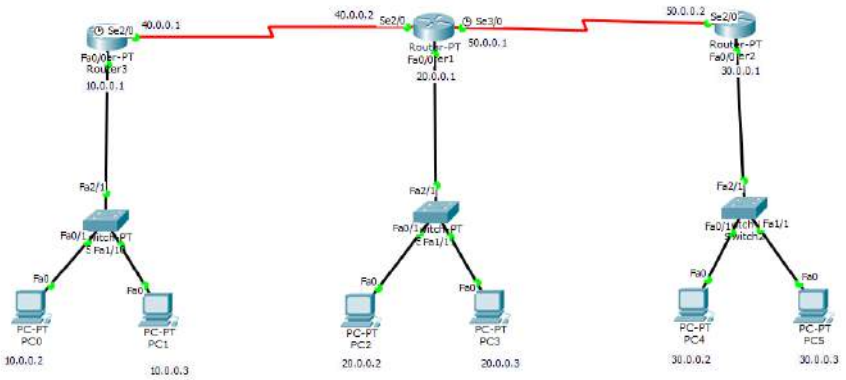
- OBSERVATION :-

1] the packet at start of transmission has the TTL = 255

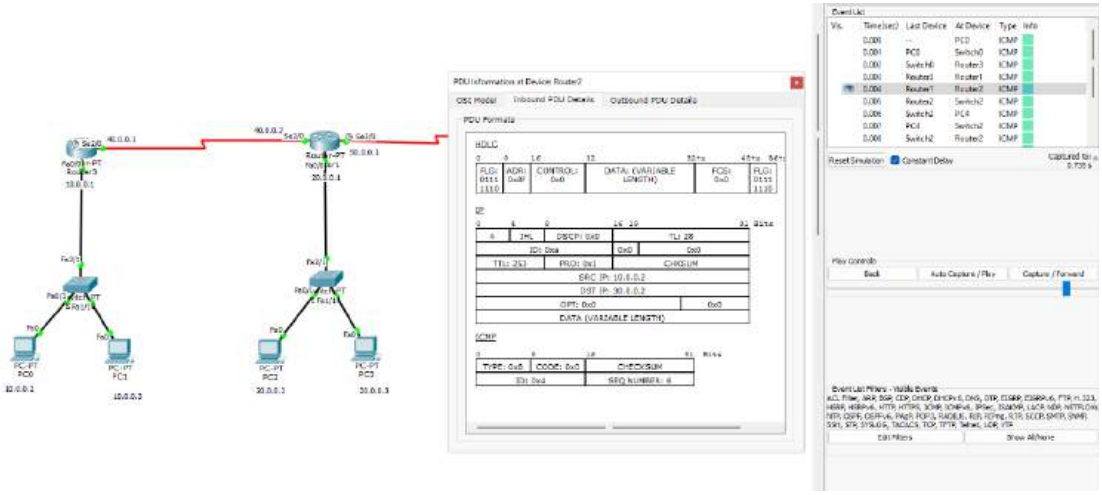
2] the TTL is seen that it is decreasing as it passes through the stages

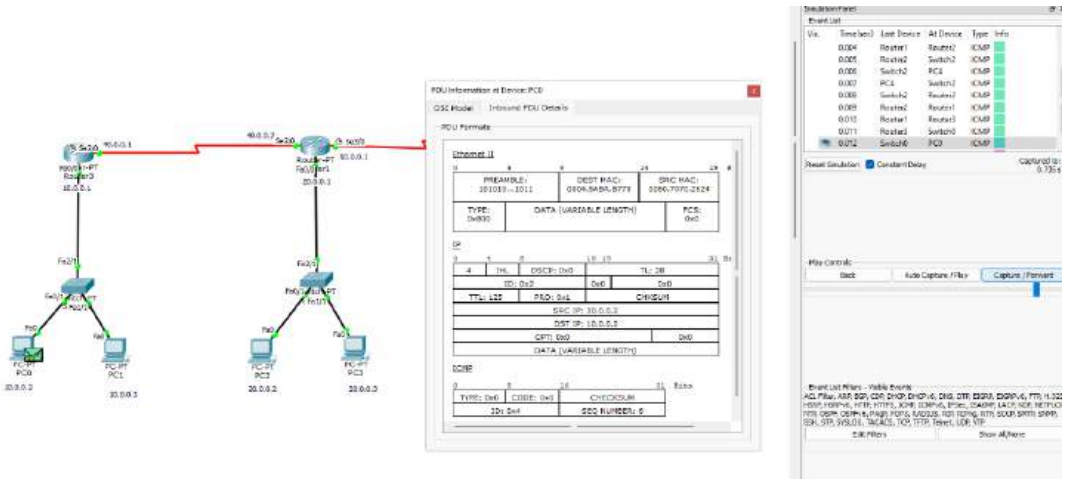
3] once the packet reaches destination TTL = 125.

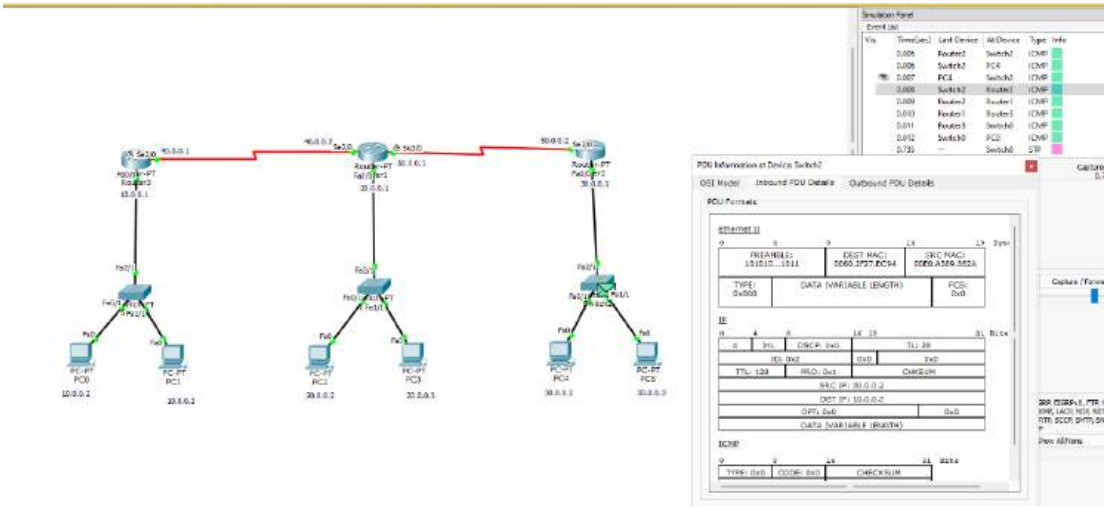
Topology:



Output:



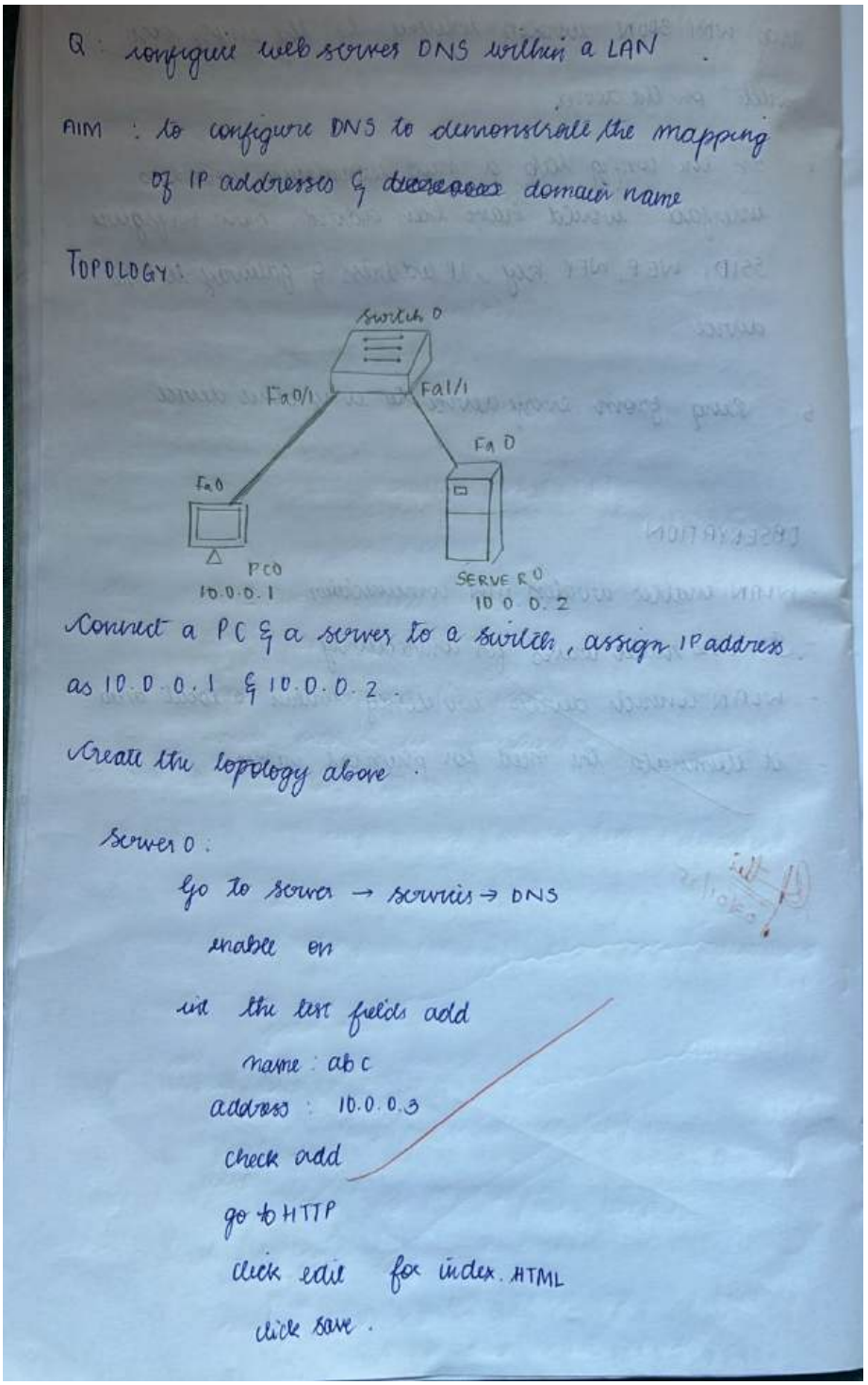




Experiment 8:

Configure Web Server, DNS within a LAN

Observation Book:



Q: configure web server DNS within a LAN

AIM: to configure DNS to demonstrate the mapping of IP addresses & domain name

TOPOLOGY:

Connect a PC & a server to a switch, assign IP address as 10.0.0.1 & 10.0.0.2.

Create the topology above.

Server 0:

go to server → service → DNS

enable on

in the list fields add

name: abc

address: 10.0.0.3

check add

go to HTTP

click edit for index.HTML

click save.

PROCEDURE :

1. Go to PC0 → desktop → Web Browser
2. search 'abc' in url box (OR)
3. search 10.0.0.2 in url box.

OUTPUT :

Cisco Packet Tracer

welcome to cisco packet tracer. opening doors to new opportunities.

Quick links :

A small page

Copyrights

image page

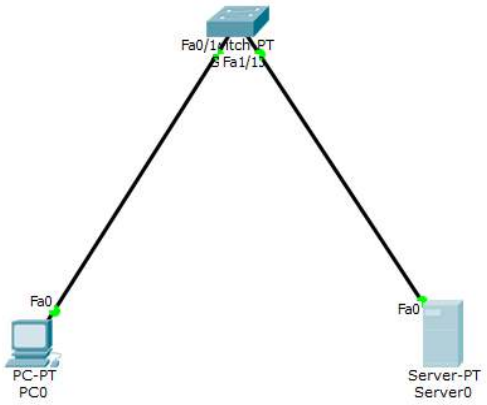
image

OBSERVATION :

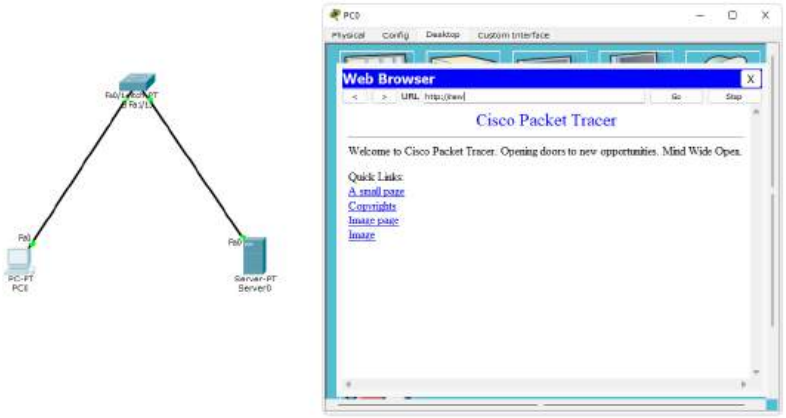
- DNS translates domain names to IP addresses.
- PC0 successfully accessed the server0 by both its IP address & configured domain name 'abc'. The configuration was successful allowing the webpage to be accessed via both methods.

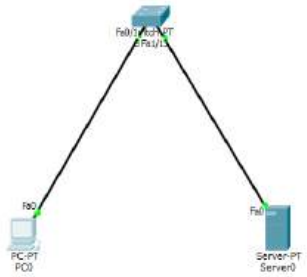
03/01/25

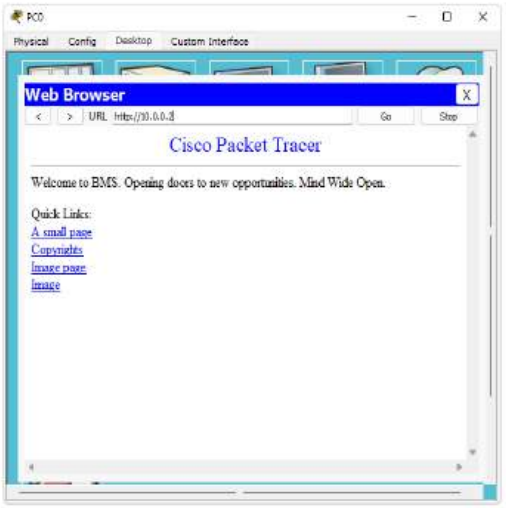
Topology:

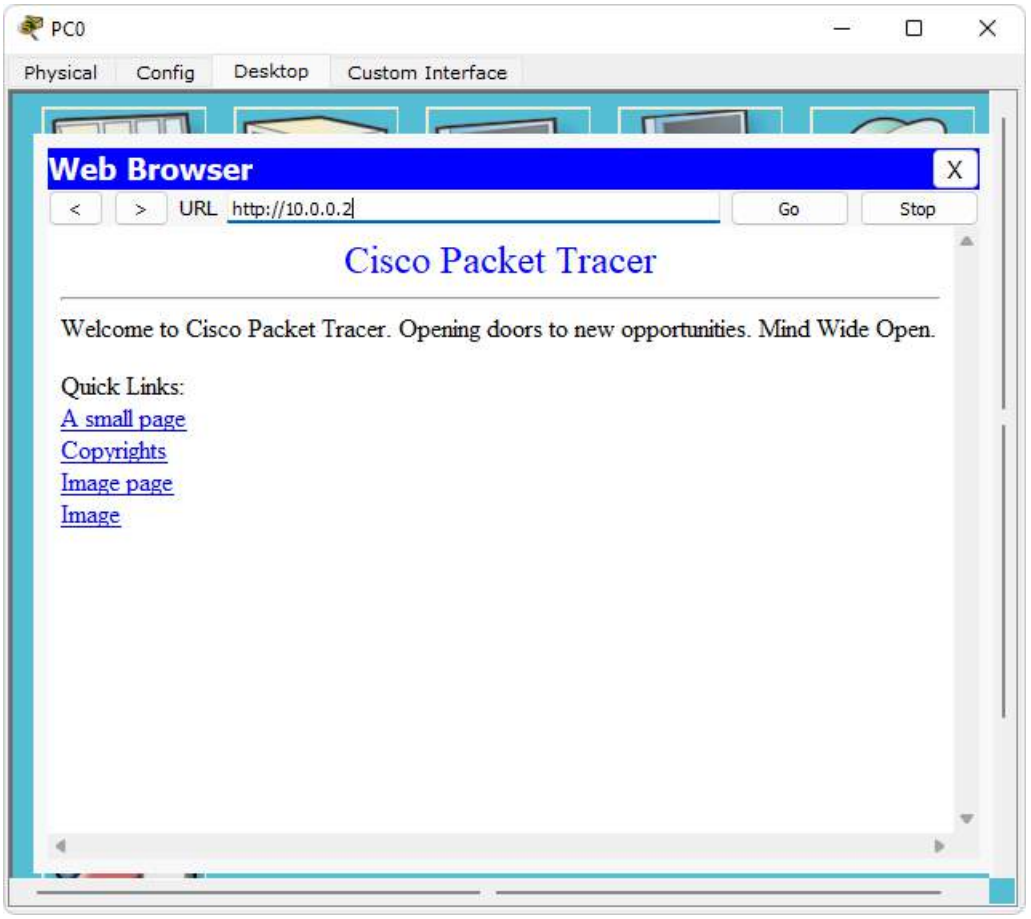


Output:




PC-PT
PC0
Server-PT
Server0

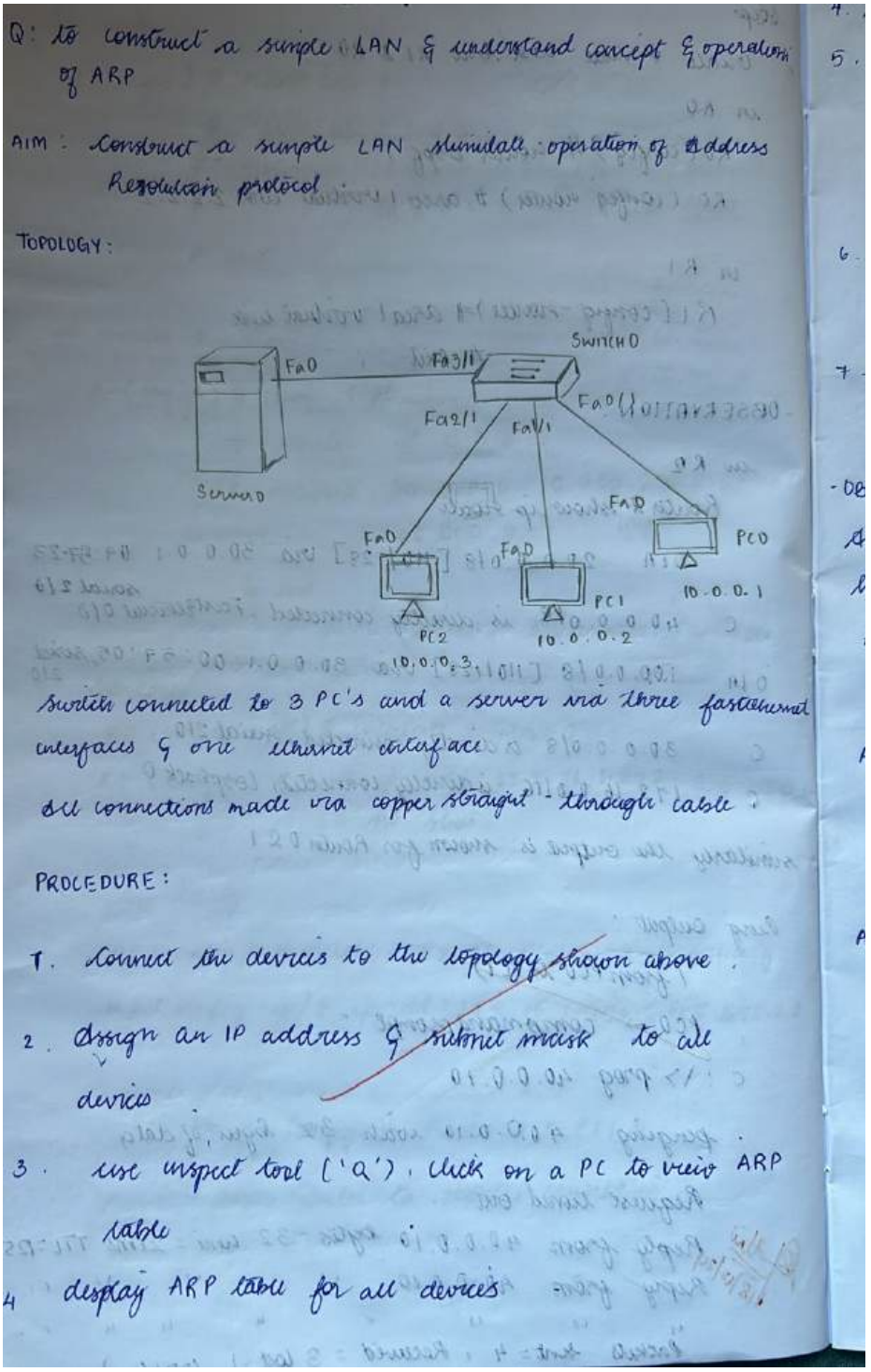

PC0
Physical
Config
Desktop
Custom Interface
Web Browser
URL
Go
Stop
Cisco Packet Tracer
Welcome to BMS. Opening doors to new opportunities. Mind Wide Open.
Quick Links:
A small page
Copyrights
Image page
Image


PC0
Physical
Config
Desktop
Custom Interface
Web Browser
URL
http://10.0.0.2
Go
Stop
Cisco Packet Tracer
Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.
Quick Links:
A small page
Copyrights
Image page
Image

Experiment 9:

To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)

Observation Book:



4. Initially ARP is empty for all

5. in the CLI of the switch:

   show mac address-table

   is given on every transaction to see how the switch learns from transitions & build the address table.

6. Use capture button in the simulation panel; go step by step so that changes in ARP is noted.

7. Observe the switch as well as nodes update the ARP table when new communication starts.

- OBSERVATION -

As the message travels from one source host to its destination host the ARP table of all devices get updated.

ARP maps an IP address to a mac address.

It ensures communication within a local network.

ARP table for PC0 (source):

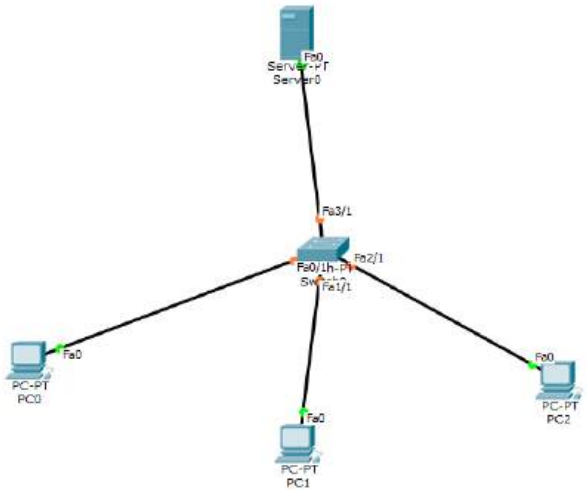
IP address	hardware address	interface
10.0.0.3	0060.2F29.2CB8	fastEthernet 0

ARP table for PC2 (destination):

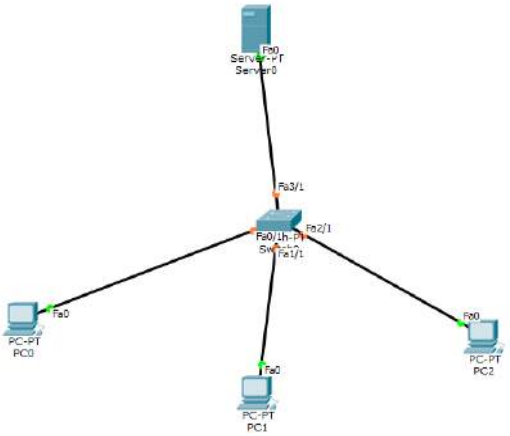
IP address	hardware Address	interface
10.0.0.1	000.0302.960B	FastEthernet 0

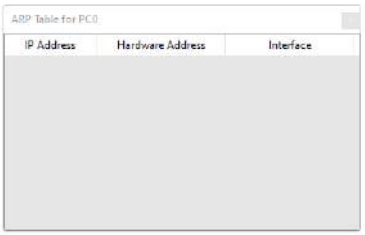
02/01/25

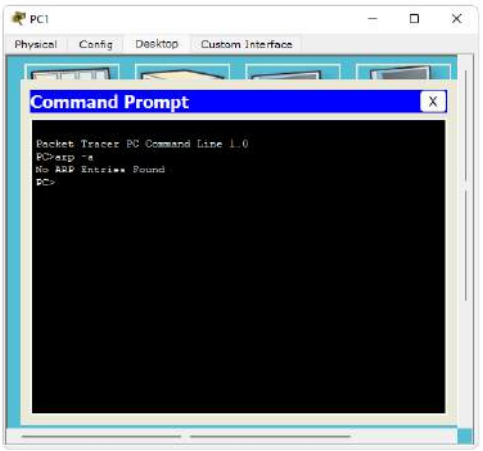
Topology:

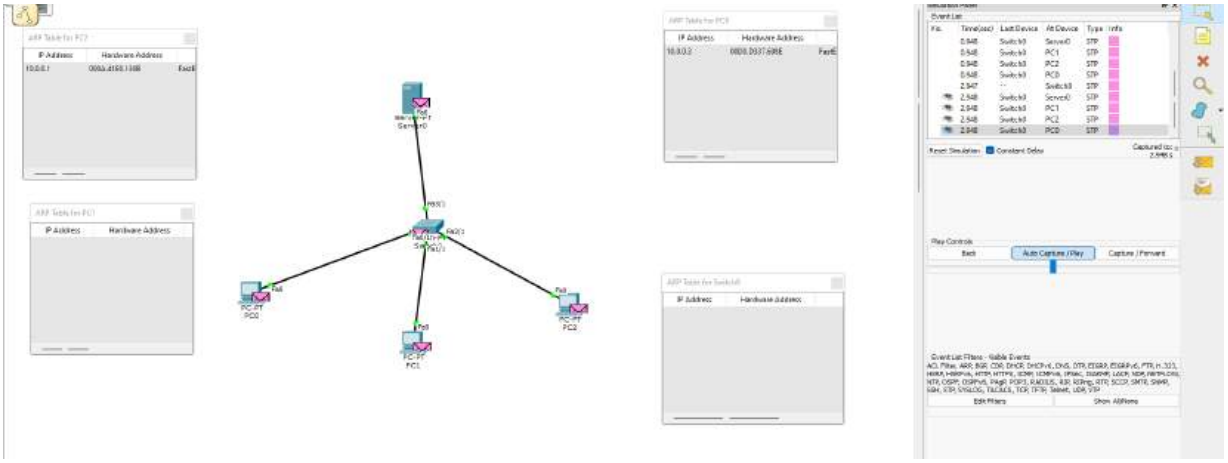


Output:





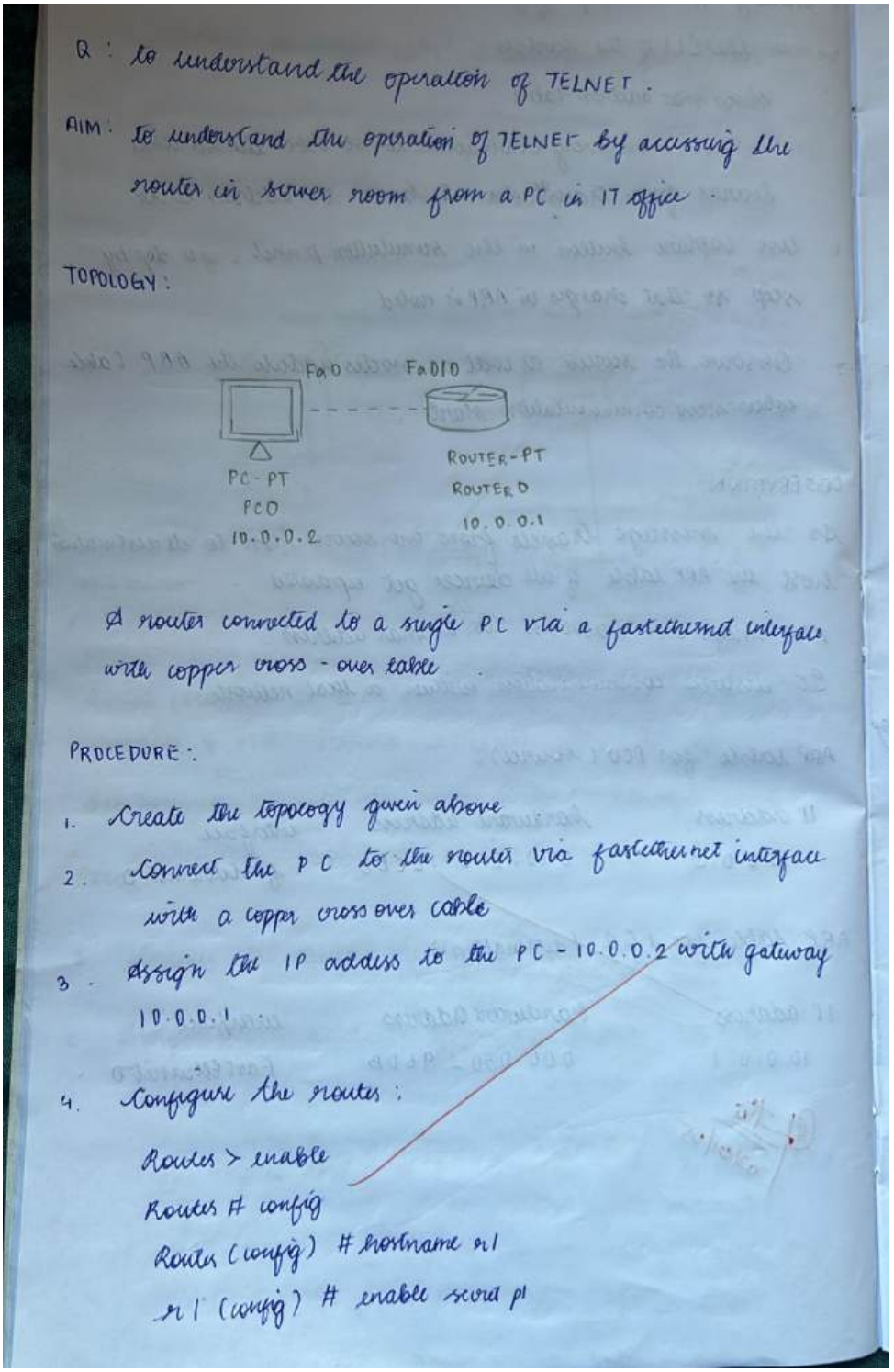




Experiment 10:

To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

Observation Book:



Q : to understand the operation of TELNET.

AIM : to understand the operation of TELNET by accessing the router in server room from a PC in IT office.

TOPOLOGY :

A router connected to a single PC via a fastethernet interface with copper cross-over table.

PROCEDURE :

1. Create the topology given above
2. Connect the PC to the router via fastethernet interface with a copper cross over cable
3. Assign the IP address to the PC - 10.0.0.2 with gateway 10.0.0.1
4. Configure the routes :

   Router > enable

   Router # config

   Router (config) # hostname r1

   r1 (config) # enable secret p1

r1(config)# interface fastethernet 0/0

r1(config-if)# ip address 10.0.0.1 255.0.0.0

r1(config-if)# no shut

r1(config-if)# line vty 0 5

r1(config-if)# login

r1(config-line)# password p0

r1(config-line)# exit

r1# wr

5. in command prompt

ping 10.0.0.1

password for user authentication is p0

password for enable is p1

OBSERVATION:

Telnet is a protocol for remote access to server

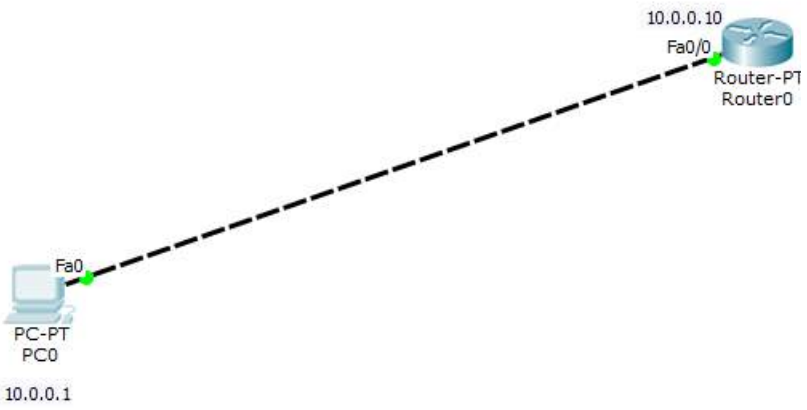
It allows command-line communication over a network.

The PC is able to send the data to the routes

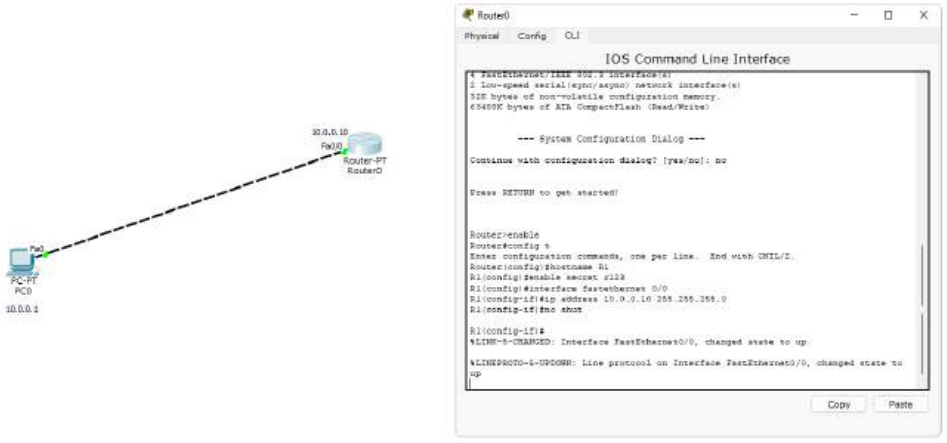
& indicates that the gateway is available & connected

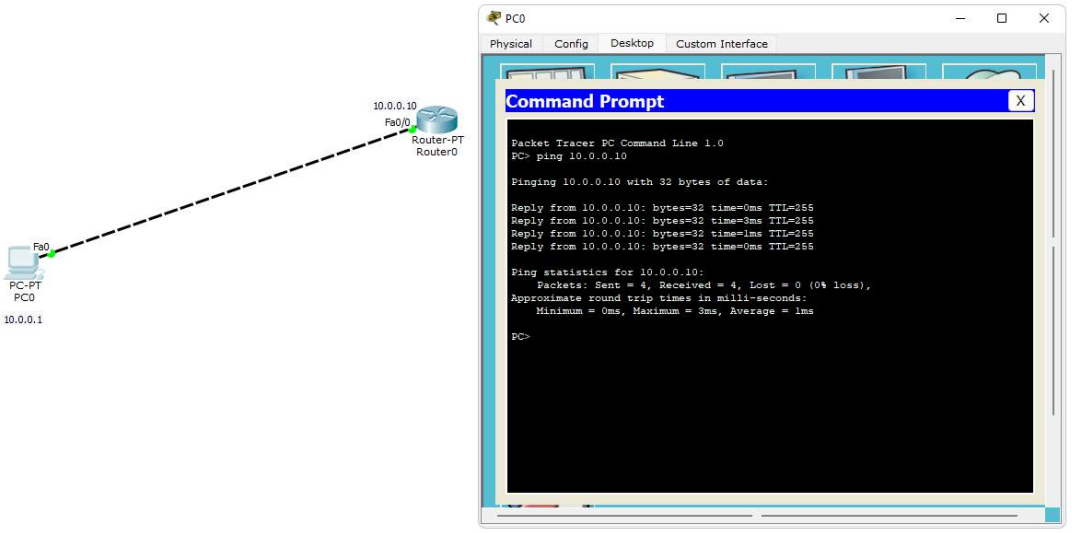
02/01/25

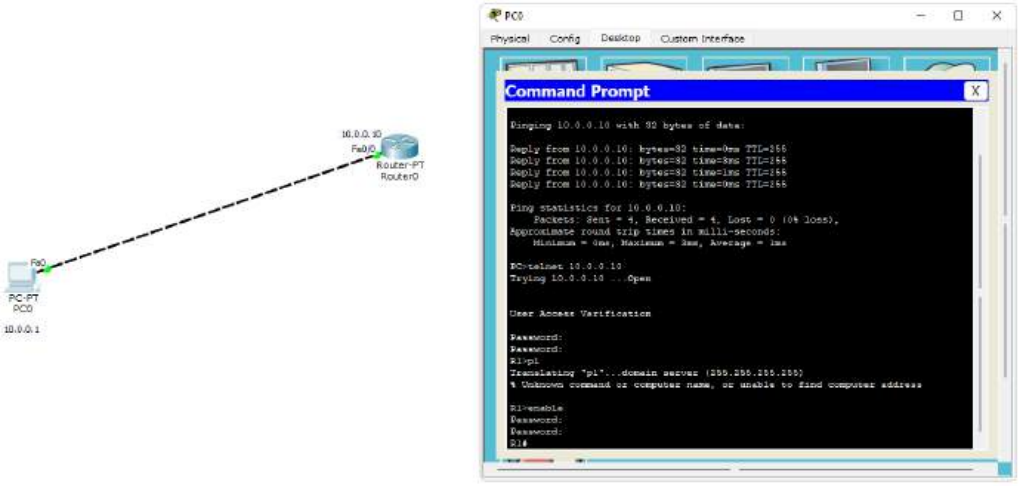
Topology:



Output:

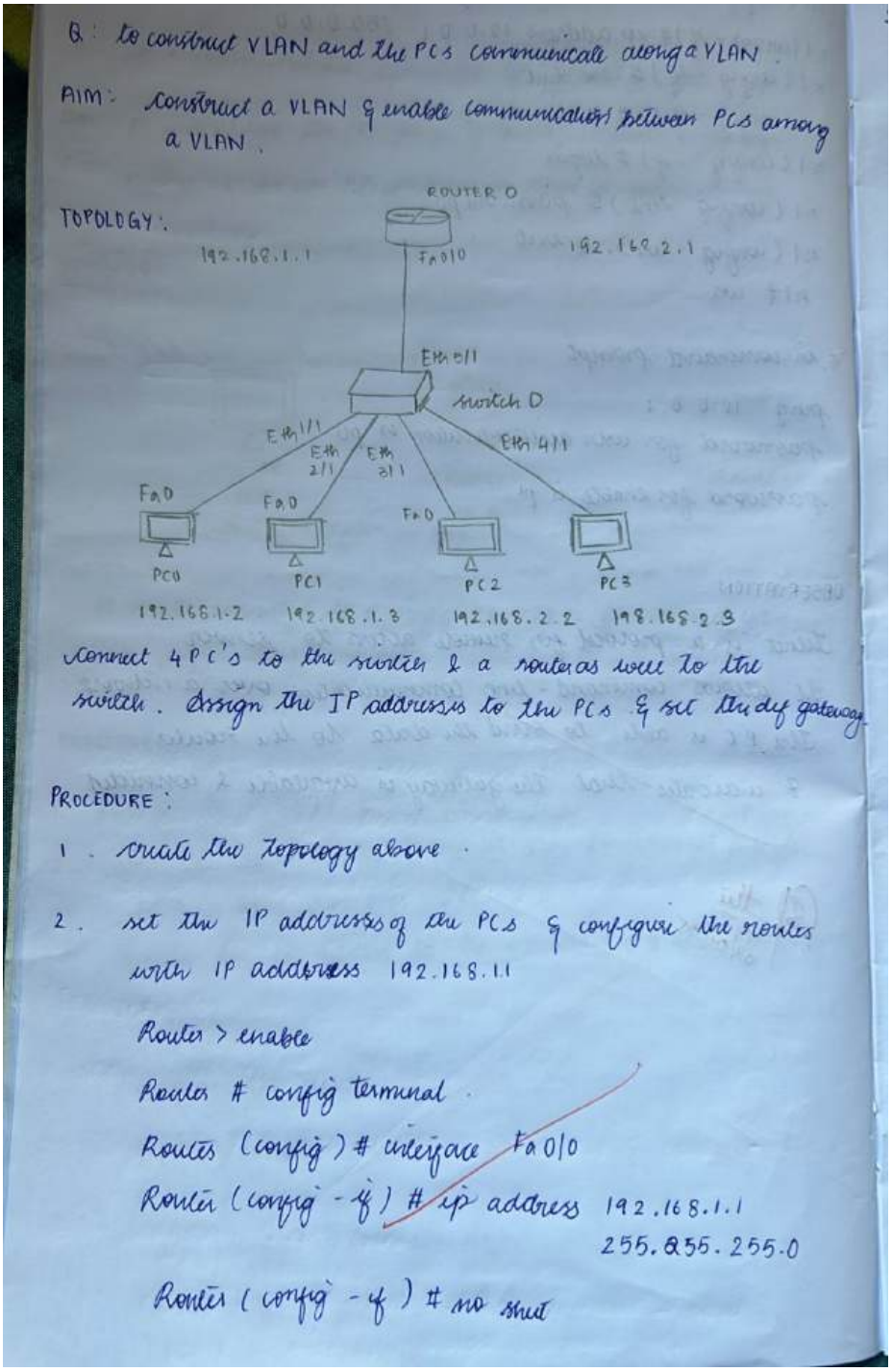



10.0.0.10
Fa0/0
Router-PT
Router0
Fa0
PC-PT
PC0
10.0.0.1
PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC> ping 10.0.0.10
Pinging 10.0.0.10 with 32 bytes of data:
Reply from 10.0.0.10: bytes=32 time=0ms TTL=255
Reply from 10.0.0.10: bytes=32 time=3ms TTL=255
Reply from 10.0.0.10: bytes=32 time=1ms TTL=255
Reply from 10.0.0.10: bytes=32 time=0ms TTL=255
Ping statistics for 10.0.0.10:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 3ms, Average = 1ms
PC>


10.0.0.10
Fa0/0
Router-PT
Router0
Fa0
PC-PT
PC0
10.0.0.1
PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Pinging 10.0.0.10 with 32 bytes of data:
Reply from 10.0.0.10: bytes=32 time=0ms TTL=255
Reply from 10.0.0.10: bytes=32 time=3ms TTL=255
Reply from 10.0.0.10: bytes=32 time=1ms TTL=255
Reply from 10.0.0.10: bytes=32 time=0ms TTL=255
Ping statistics for 10.0.0.10:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 3ms, Average = 1ms
PC>telnet 10.0.0.10
Trying 10.0.0.10 ...Open
User Access Verification
Password:
Password:
R1>p1
Translating "p1"...domain server (255.255.255.255)
% Unknown command or computer name, or unable to find computer address
R1>enable
Password:
Password:
R1#

Experiment 11:
To construct a VLAN and make the PC's communicate among a VLAN

Observation:



3. in the switch, go to config tab & select VLAN database
4. set the VLAN number & VLAN name.
select the interface i.e fastethernet 5/1 & make it the trunk
VLAN tracking allows switches to forward frame from different VLAN over a single link called trunk.

5. this is done by adding additional header information called tag to the ethernet frame

6. look into the interfaces of the switches with 2 NEW VLAN system
config tab of the router select VLAN DATABASE &
enter number & name of VLAN created.

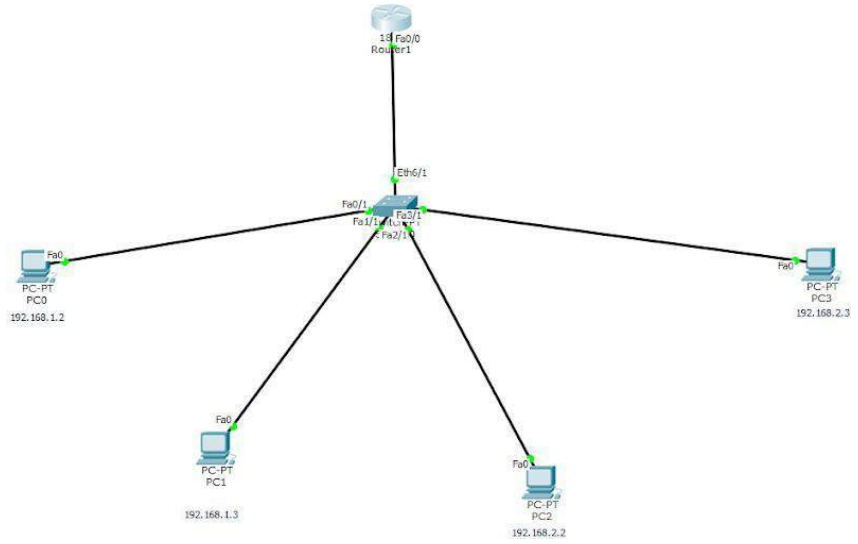
Router (vlan)# exit
Router # config terminal
Router (config)# interface fastethernet 0/0.1
Router (config-subif)# encapsulation dot1q 2.
Router (config-subif)# ip address 192.168.2.1 255.255.255.
Router (config-subif)# no shut
Router (config-subif)# exit
Router (config)# exit

OBSERVATION

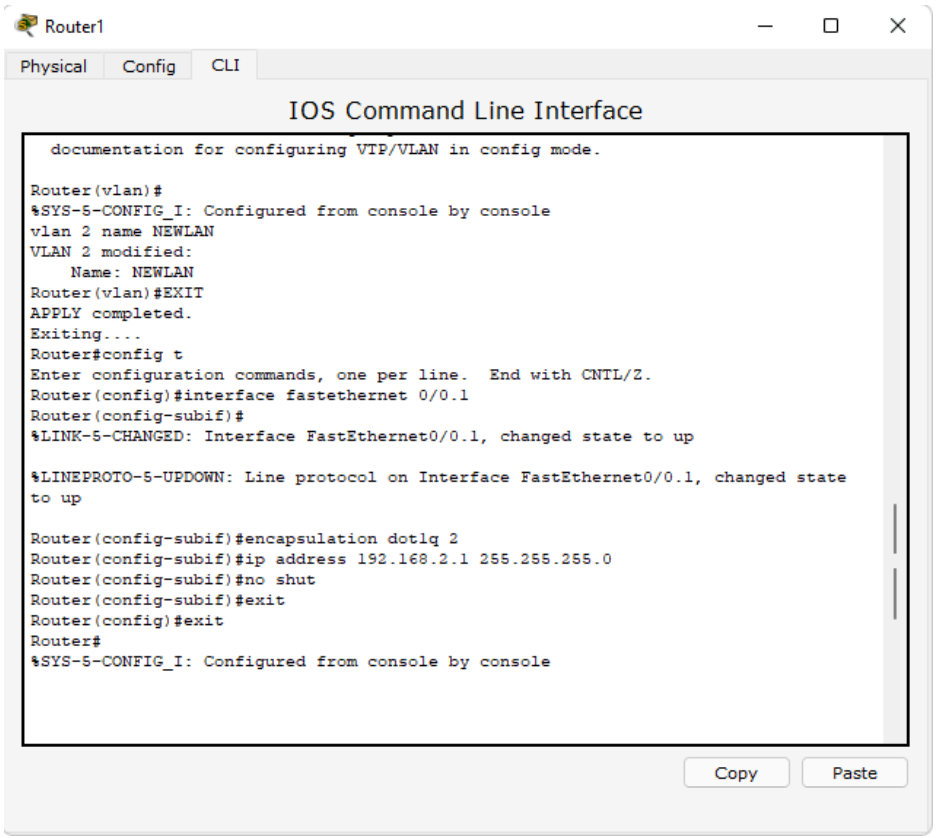
A VLAN segments a network into vertical grps
It enhances security & reduces broadcast traffic. On pinging over the VLAN the PCs are able to communicate

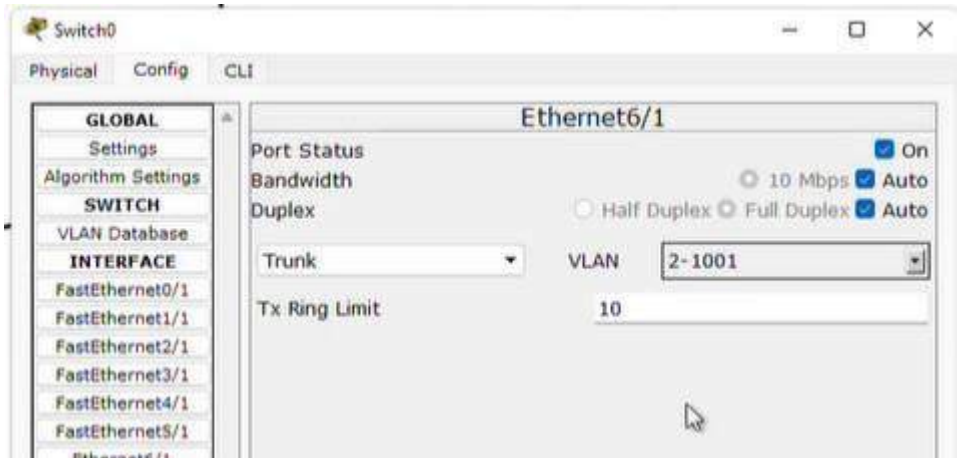
Thu 02/01/25

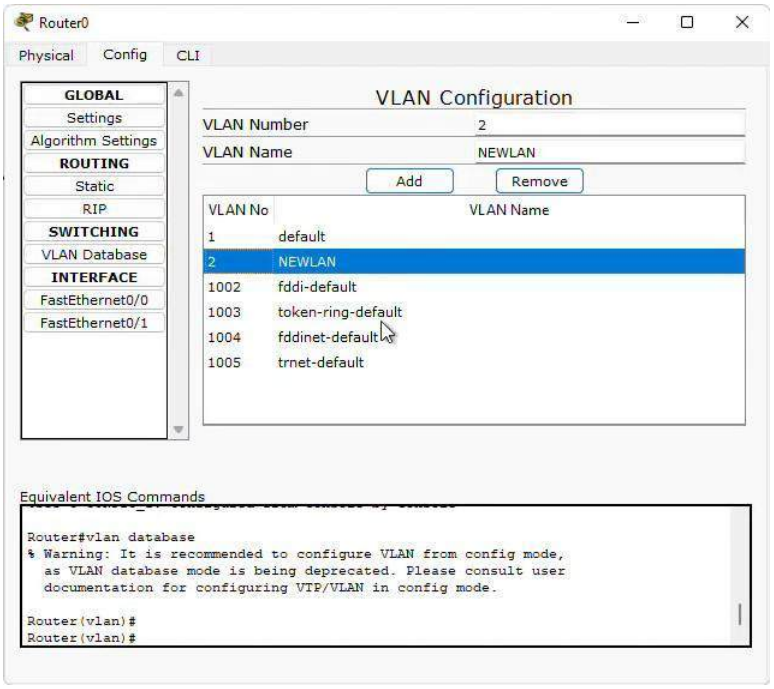
Topology:



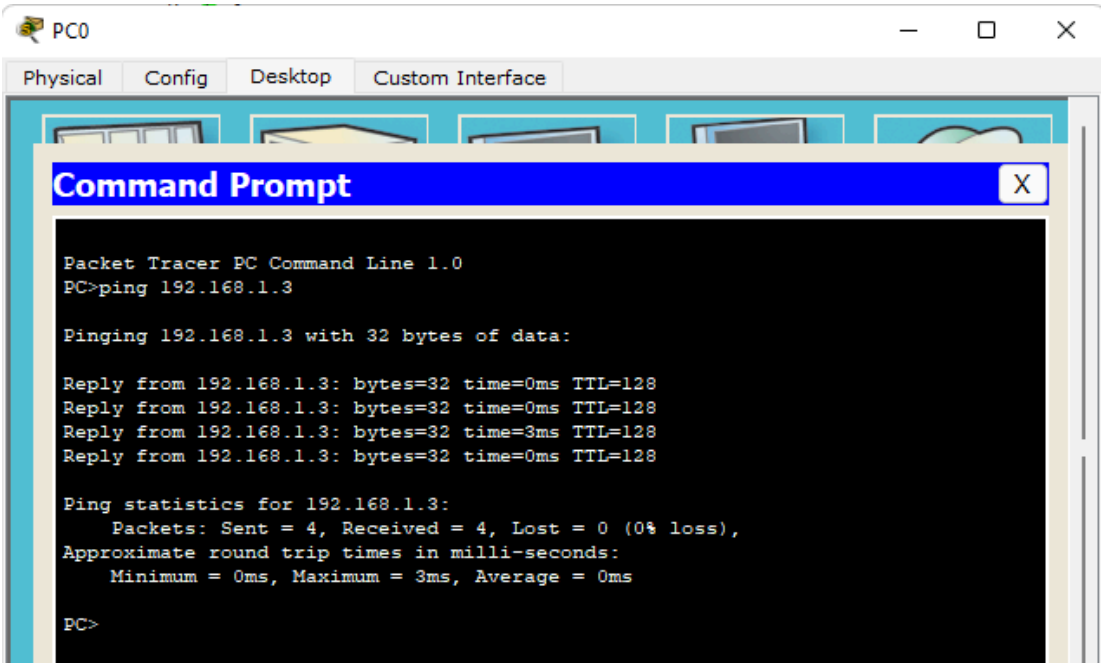
Output:





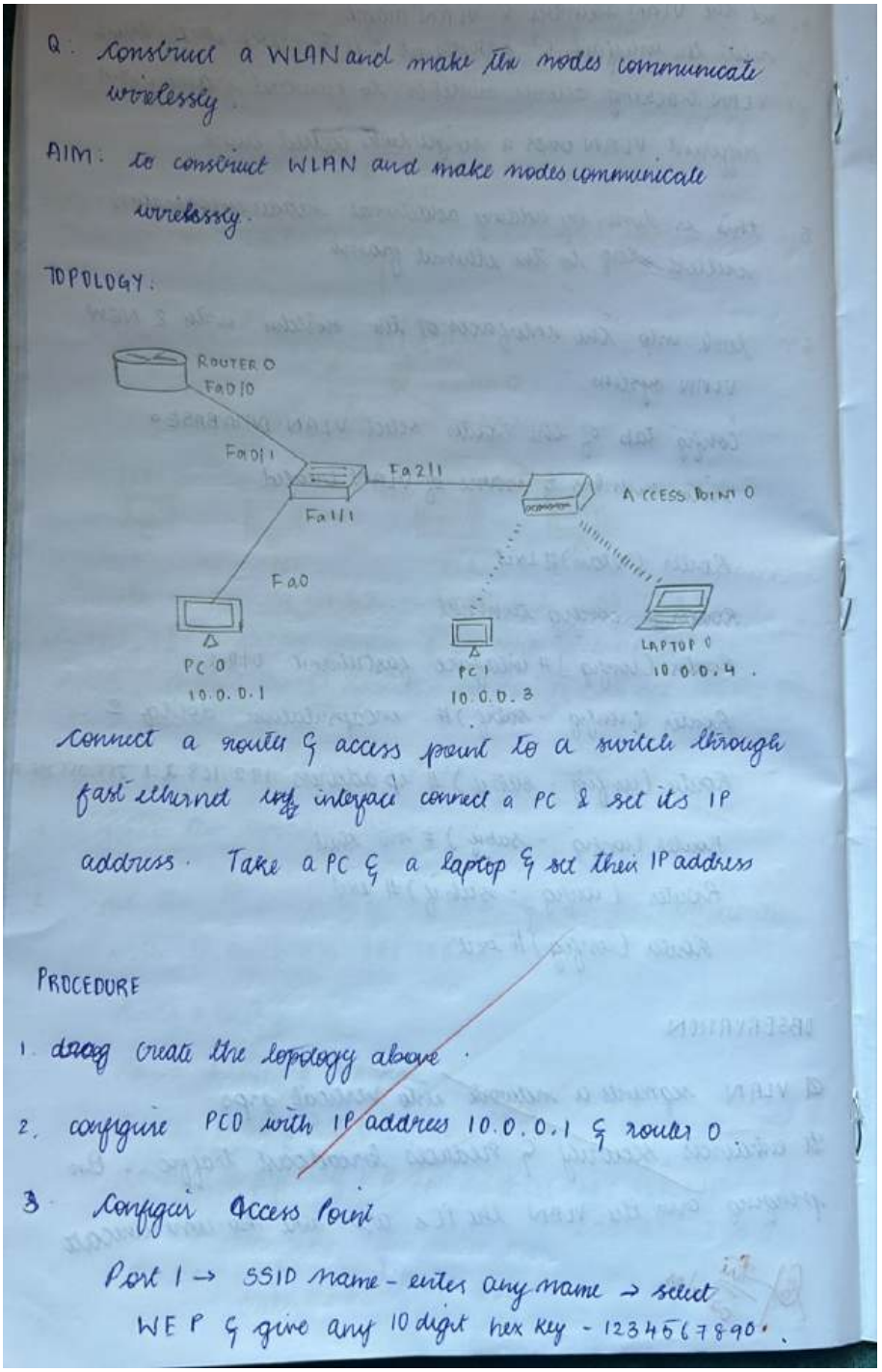



```
%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed state to
up
exit
Router(config)#
Router(config)#exit
Router#vlan database
% Warning: It is recommended to configure VLAN from config mode,
  as VLAN database mode is being deprecated. Please consult user
  documentation for configuring VTP/VLAN in config mode.

Router(vlan)#
%SYS-5-CONFIG_I: Configured from console by console
vlan 2 name NEWLAN
VLAN 2 modified:
    Name: NEWLAN
Router(vlan)#EXIT
```



Experiment 12:
To construct a WLAN and make the nodes communicate wirelessly

Observation Book:



4. Configure off the device drag the substring PT-HOST-NM-1FD drag WMP 300N wireless interface to the empty port switch on the device.

5. In the config tab a new ~~wireless~~ wireless interface would have been added. Now configure SSID, WEP, WEP key, IP address & gateway to the device.

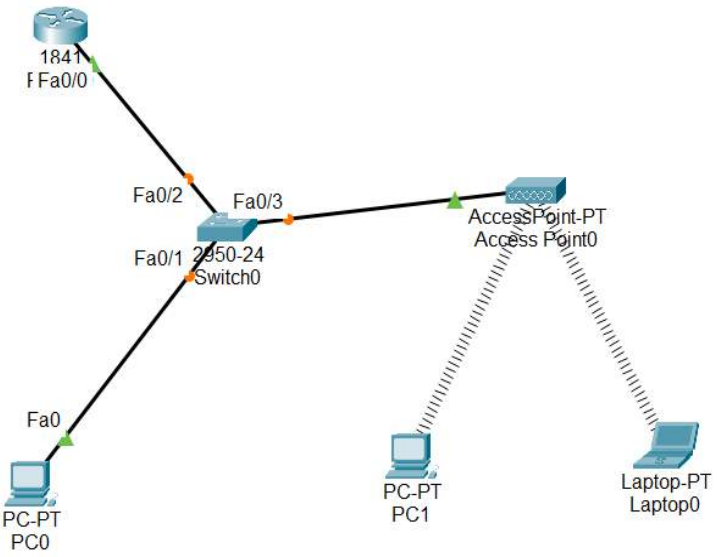
6. Ping from every device to every other device.

OBSERVATION

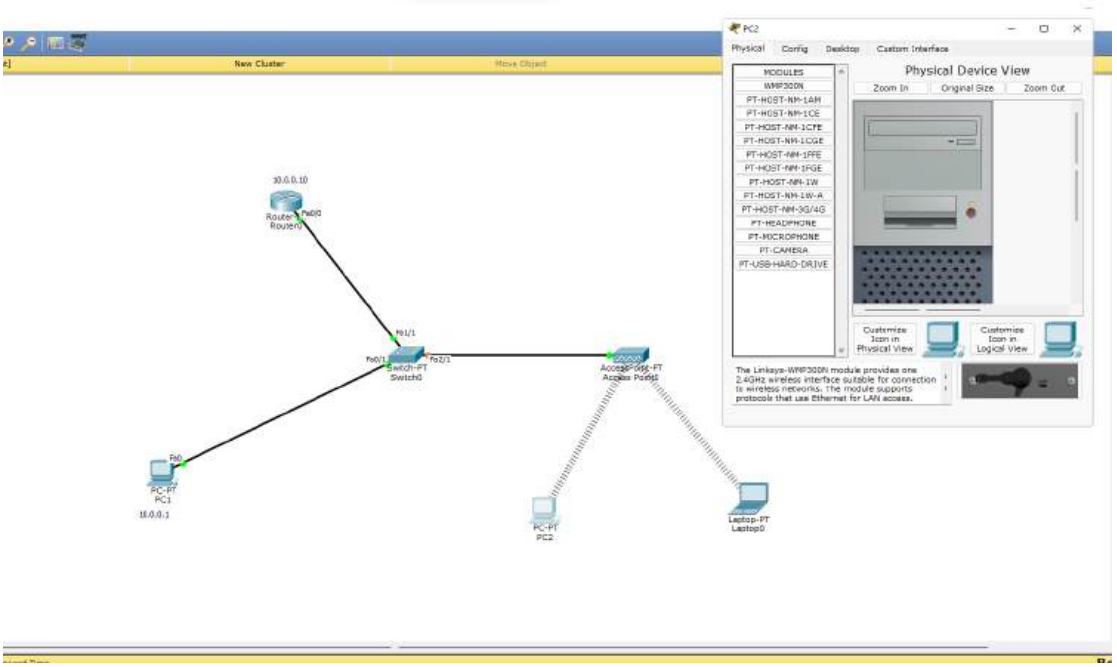
- WLAN enables wireless n/w communication.
- It uses radio waves for connectivity
- WLAN connects devices wirelessly within a local area.
- It eliminates the need for physical cables.

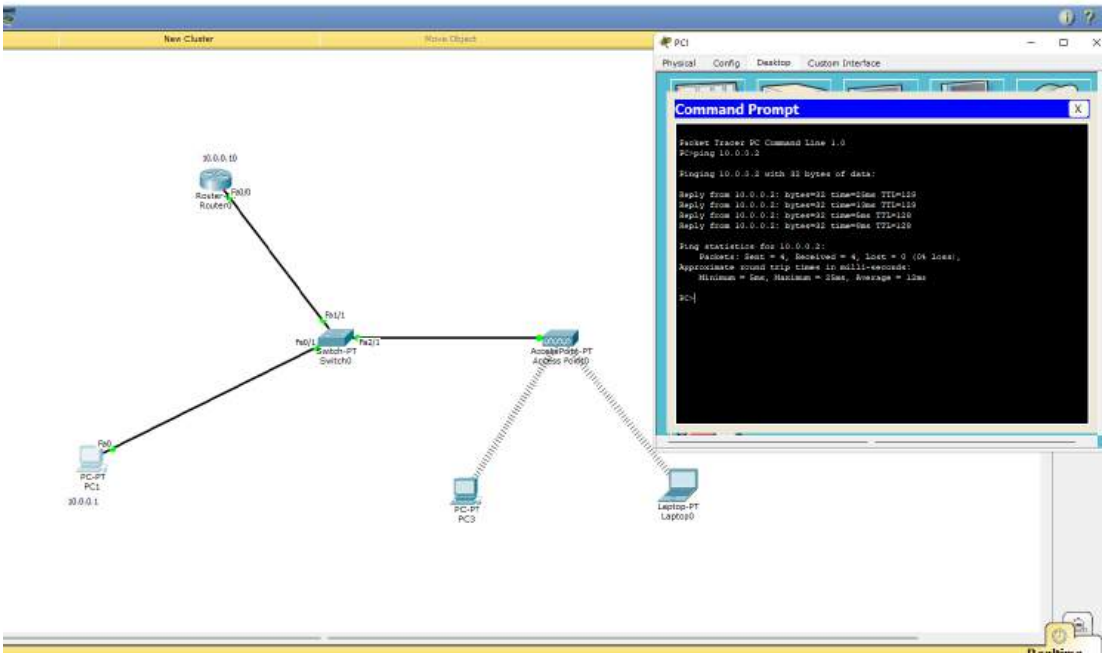
03/01/25

Topology:



Output:



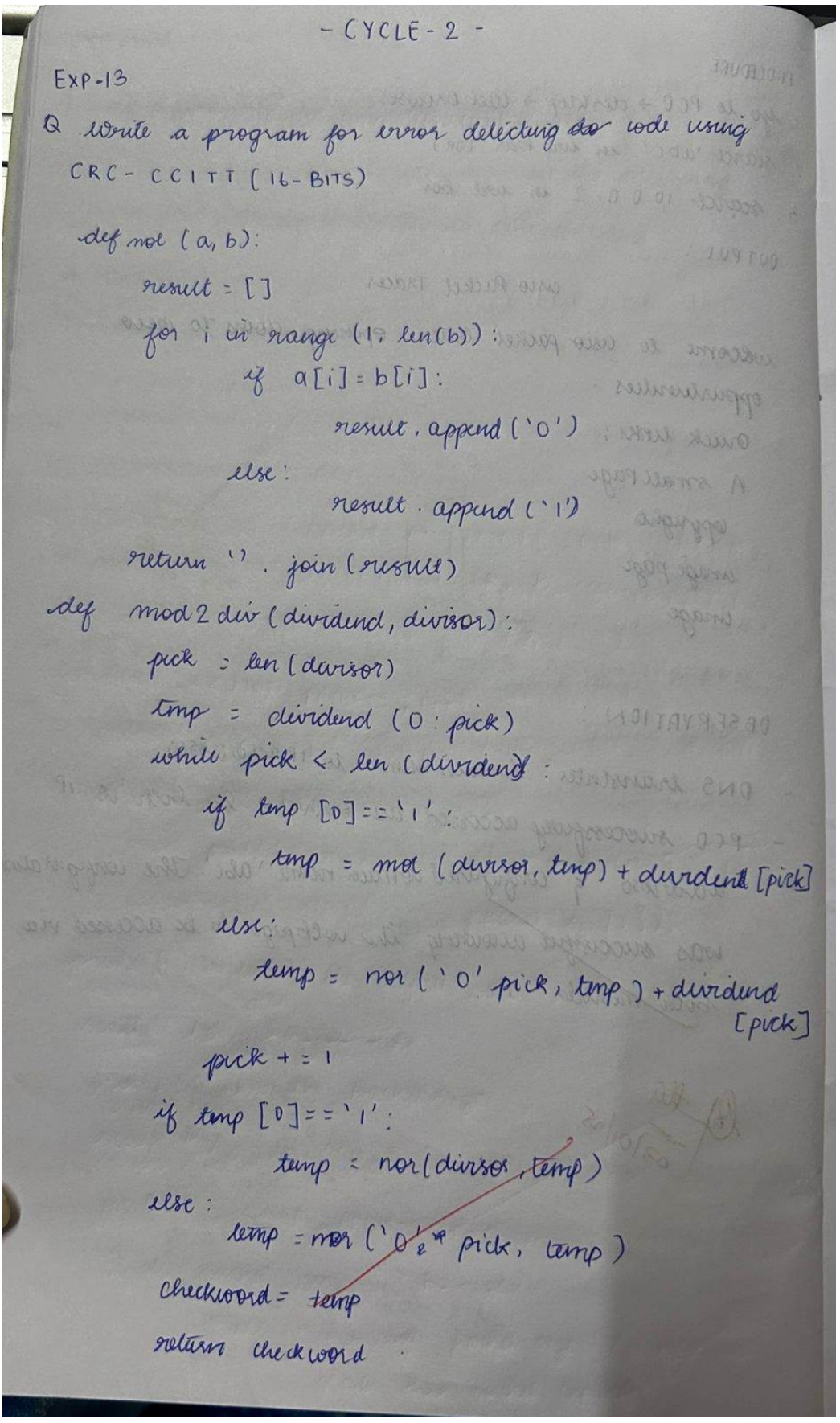

New Cluster
10.0.0.10
Router0
Fa0/0
Fa1/1
Fa0/1
Fa2/1
Switch-PT
Switch0
Access Point-PT
Access Point0
Fa0
PC-PT
PC1
10.0.0.1
PC-PT
PC3
Laptop-PT
Laptop0
PC1
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.2
Pinging 10.0.0.2 with 32 bytes of data:
Reply from 10.0.0.2: bytes=32 time=26ms TTL=128
Reply from 10.0.0.2: bytes=32 time=10ms TTL=128
Reply from 10.0.0.2: bytes=32 time=6ms TTL=128
Reply from 10.0.0.2: bytes=32 time=6ms TTL=128
Ping statistics for 10.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 6ms, Maximum = 26ms, Average = 12ms
PC>

Cycle-2

Program 1:

Write a program for error detecting code using CRC-CCITT (16-bits)

Observation:



- CYCLE - 2 -

EXP-13

Q Write a program for error detecting code using CRC-CCITT (16-BITS)

```
def xor (a, b):
    result = []
    for i in range (1, len(b)):
        if a[i] = b[i]:
            result . append ('0')
        else:
            result . append ('1')
    return '' . join (result)
def mod2 div (dividend, divisor):
    pick = len (divisor)
    tmp = dividend (0 : pick)
    while pick < len (dividend):
        if tmp [0] == '1':
            tmp = xor (divisor, tmp) + dividend [pick]
        else:
            temp = xor ('0' pick, tmp ) + dividend
                                          [pick]
        pick + = 1
    if tmp [0] == '1':
        temp = xor(divisor, temp)
    else:
        temp = xor ('0' * pick, temp )
    checkword = temp
    return checkword
```

```
def encode (data, key):
        l - key = len(key)
        appended_data = data + '0' * (len-key -1)
        remainder = mod 2 div (appended_data. key)
        codeword = data + remainder
        print ("Remainder : ", remainder)
        print ('encoded data [data + remainder] ", codeword)
        return codeword.

def decode-data (encoded-data, Key)
        remainder = mod 2 div (encoded_data, Key).
        print ("Remainder after decoding : ", remainder)
        if '1' not in remainder:
            print ("no error detected in received data")
        else
            print ("error detected in received data")

data = "10010010001001 00"
key = "1101"
encoded-data = encode (data, Key)
decoded_data = decode_data (encoded-data, key)
```

OUTPUT :

remainder = 11

encoded_data (data + remainder) = 100100100010010011

no error detected in received data.

thi
02/01/25

Code:

```
def crc_ccitt_16_bitstream(bitstream: str, poly: int = 0x1021, init_crc: int =
0xFFFF) -> int:
    """
    Calculate the 16-bit CRC-CCITT checksum for a given binary string.
    """
    crc = init_crc
    for bit in bitstream:
        crc ^= int(bit) << 15  # Align the bit with CRC's uppermost bit
        for _ in range(8):  # Process each bit
            if crc & 0x8000:  # Check if the leftmost bit is set
                crc = (crc << 1) ^ poly
            else:
                crc <<= 1
            crc &= 0xFFFF  # Ensure CRC remains 16-bit
    return crc


def append_crc_to_bitstream(bitstream: str) -> str:
    """
    Append the calculated 16-bit CRC to the given bitstream.
    """
    crc = crc_ccitt_16_bitstream(bitstream)
    crc_bits = f"{crc:016b}"  # Convert CRC to a 16-bit binary string
    return bitstream + crc_bits


def verify_crc_bitstream(bitstream_with_crc: str) -> bool:
    """
    Verify the CRC of the given bitstream with CRC appended.
    """
    if len(bitstream_with_crc) < 16:
        return False  # Not enough bits to contain CRC
    data, received_crc = bitstream_with_crc[:-16], bitstream_with_crc[-16:]
    calculated_crc = crc_ccitt_16_bitstream(data)
    return calculated_crc == int(received_crc, 2)


# Main Program
if __name__ == "__main__":
    # User input for original bitstream
    message_bits = input("Enter the original bitstream (e.g., 11010011101100):
").strip()

    # Validate input
    if not all(bit in "01" for bit in message_bits):
        print("Invalid input. Please enter a binary bitstream (e.g.,
11010011101100).")
    else:
        # Calculate and append CRC
        bitstream_with_crc = append_crc_to_bitstream(message_bits)
        print(f"Transmitted bitstream with CRC: {bitstream_with_crc}")

        # User input for received bitstream
        user_bitstream = input("Enter the received bitstream for verification:
").strip()

        # Validate received input
        if not all(bit in "01" for bit in user_bitstream):
            print("Invalid input. Please enter a valid binary bitstream.")
        elif len(user_bitstream) < 16:
```

```
            print("Invalid input. Received bitstream must include at least 16
bits for CRC.")
        else:
            # Verify CRC
            is_valid = verify_crc_bitstream(user_bitstream)
            if is_valid:
                print("No errors detected. CRC valid.")
            else:
                print("Error detected! CRC invalid.")
```

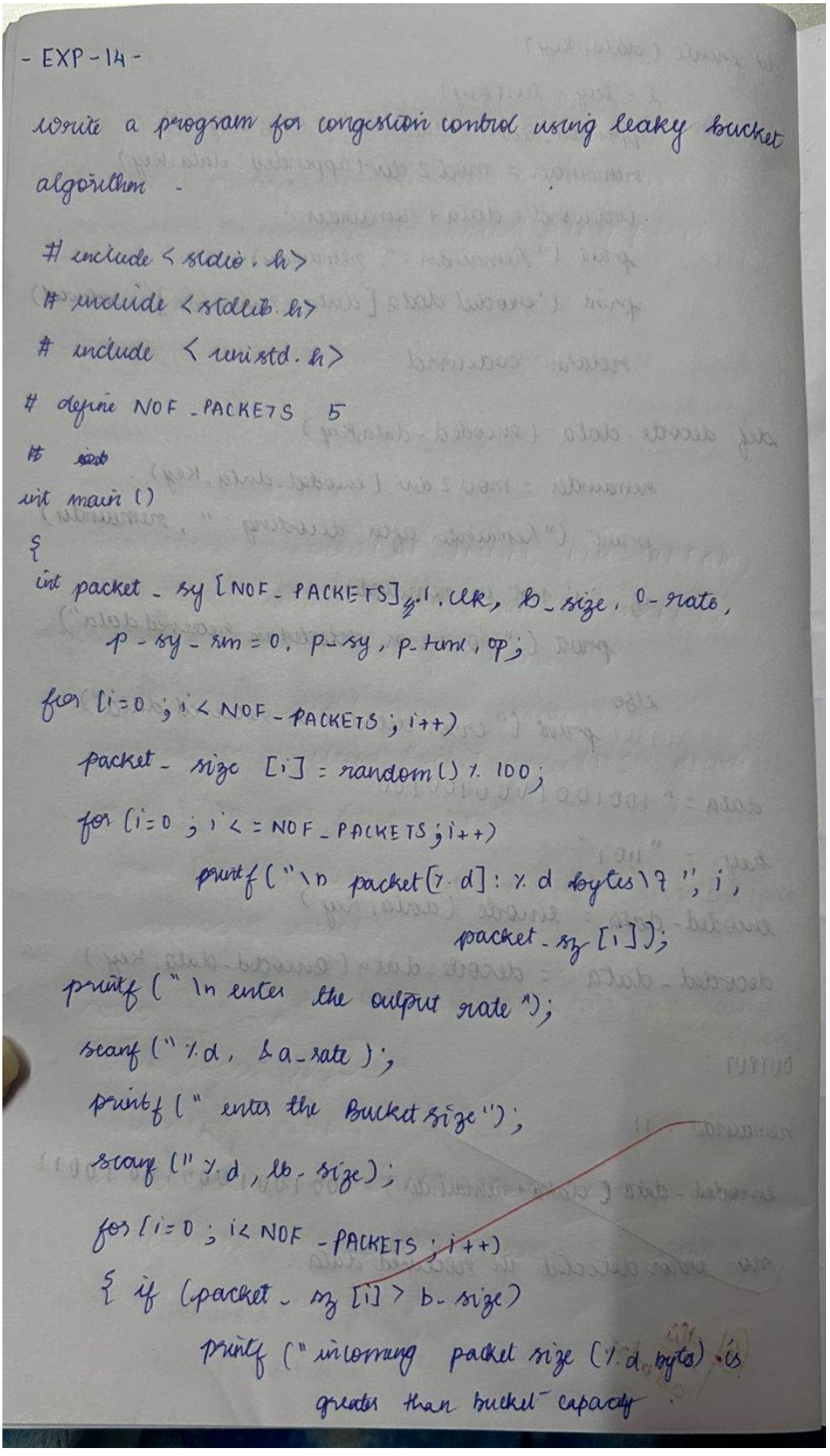
Output:

```
Enter data to be transmitted: 1001100
Enter the Generating polynomial: 100001011

---------------------------------------
Data padded with n-1 zeros: 100110000000000
CRC or Check value is: 0100010
Final data to be sent: 10011000100010
---------------------------------------

Enter the received data: 10011000100011

-----------------------------
Data received: 10011000100011
Error detected
```

Program 2:

Write a program for congestion control using Leaky bucket algorithm.

Observation:



- EXP - 14 -

write a program for congestion control using leaky bucket algorithm.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#define NOF_PACKETS 5
#
int main()
{
int packet_sz[NOF_PACKETS], i, clk, b_size, o_rate,
    p_sz_rm=0, p_sz, p_time, op;

for(i=0; i<NOF_PACKETS; i++)
   packet_size[i] = random() % 100;
   for(i=0; i<=NOF_PACKETS; i++)
          printf("\n packet[%d]: %d bytes\t", i,
                                packet_sz[i]);
  printf("\n enter the output rate");
   scanf("%d", &o_rate);
   printf("enter the Bucket size");
   scanf("%d", &b_size);
   for(i=0; i<NOF_PACKETS; i++)
   { if (packet_sz[i] > b_size)
          printf("incoming packet size (%d bytes) is
                 greater than bucket capacity
```

bucket

(%d bytes) is greater
than bucket capacity);

else
printf("bucket capacity exceeded packets rejected")

```
else {
p->sz = rmt = packet_sz[i];
printf("incoming packet size : %d", packet_sz[i]);
printf("\n Bytes remaining to transmit ! %d",
                                p->sz - rm);

p->time = random() % 10;
printf("time for transmission : %d unit", p->time);
for (clk = 10; clk <= p->time, clk += 10)
while (p->sym - rm > 0)
  { sleep(1);
    if (p->size - rm)
      { if (p->sz - rm <= 10_rate)
          op = p->sz - rm, p->sz - rm = 0;
    else
          sp = 0_rate, p->sz - rm = 0 - rate
          printf( Packet of size %d transmitted sp);
  }
  else
    printf("no packets to transmit!");
}
}
}
}
}
```

03/01/25

Code:

```
storage=0
noofqueries=int(input("Enter no of queries:"))
bucketsize=int(input("Enter bucket size:"))
inputpktsize=int(input("Enter input packet size:"))
outputpktsize=int(input("Enter output packet size:"))
for i in range(0,noofqueries):
    sizeleft=bucketsize-storage
    if inputpktsize<=sizeleft:
        storage+=inputpktsize
    else:
        print("Packet loss=", inputpktsize)
    print(f"Bucket size={storage}out of bucket size={bucketsize}")
    storage-=outputpktsize
```

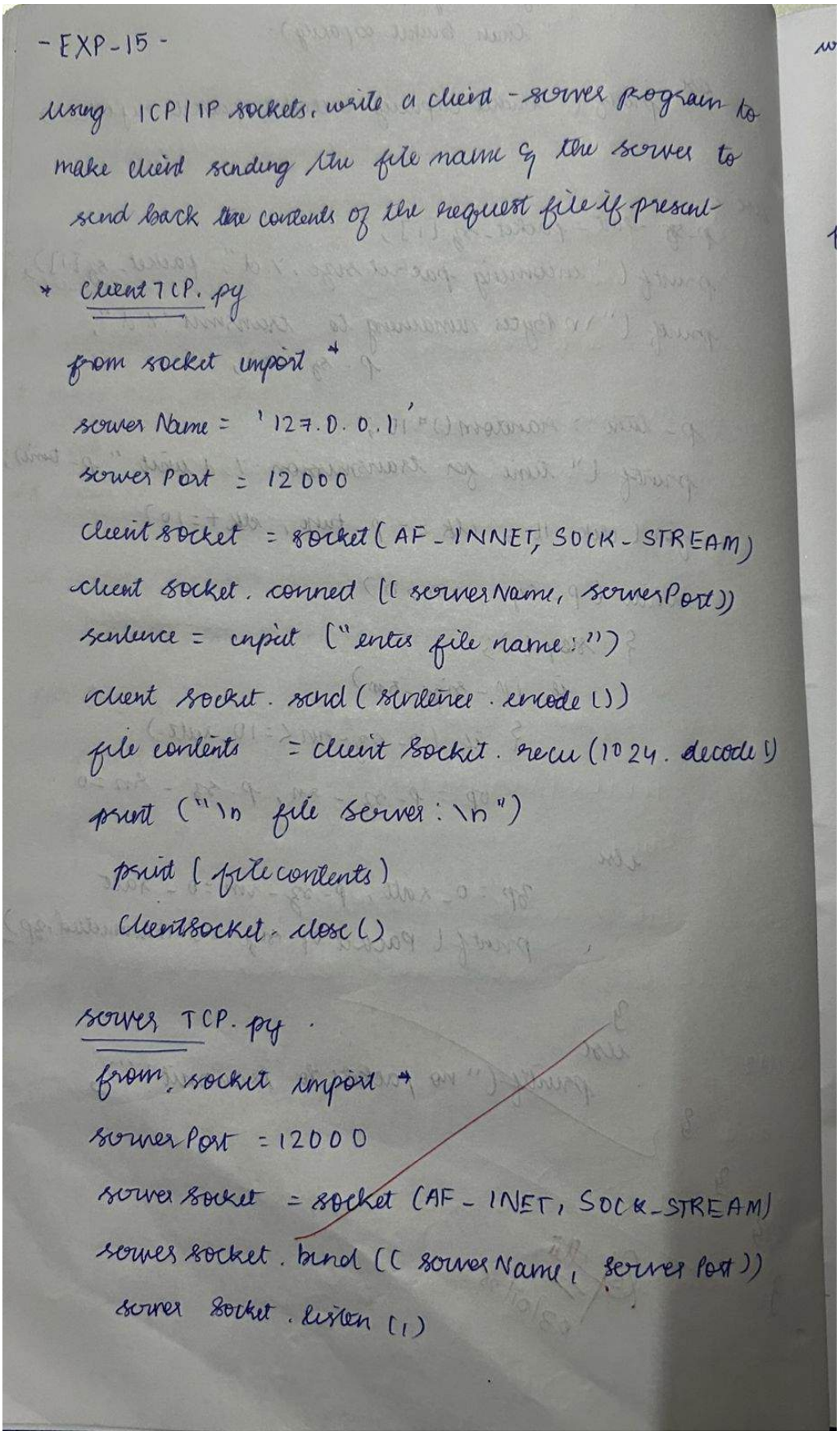
Output:

```
Enter initial packets in the bucket: 0
Enter total no. of times bucket content is checked: 4
Enter total no. of packets that can be accommodated in the bucket: 10
Enter no. of packets that enters the bucket at a time: 4
Enter no. of packets that exits the bucket at a time: 1
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:

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.

Observation:



- EXP-15 -

Using TCP/IP sockets, write a client-server program to make client sending the file name & the server to send back the contents of the request file if present

* Client TCP.py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INNET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence = input("enter file name:")
clientSocket.send(sentence.encode())
fileContents = clientSocket.recv(1024.decode())
print("\n file Server:\n")
print(fileContents)
ClientSocket.close()
```

ServerTCP.py

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

```
while 1 :
        print ("server ready to receive")
         connection . socket . add = serversocket . accept ( )
         sentence = connection socket . receive (1024) . decode .

file = open ( sentence, 'r').
  l = file . read (1024)
  connection socket . send ( l . encode ( ))
  print ("\n send contents of + sentence)
   file . close ( )
    connection socket . close ( ) .
```

OUTPUT :

the server is ready to receive
sent contents of server TCP . py
the server is ready to receive

enter file name : serverTCP . py .
Reply from server :

03/01/25

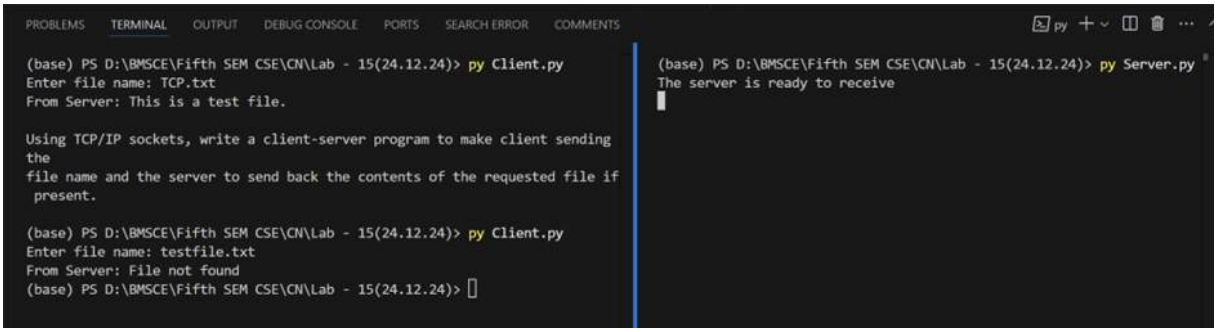
Servertcp.py

```
from socket import *
serverName="127.0.0.1"
serverPort = 14000
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()
```

Clienttcp.py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 14000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))
sentence = input("\nEnter file name: ")

clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print ('\nFrom Server:\n')
print(filecontents)
clientSocket.close()
```

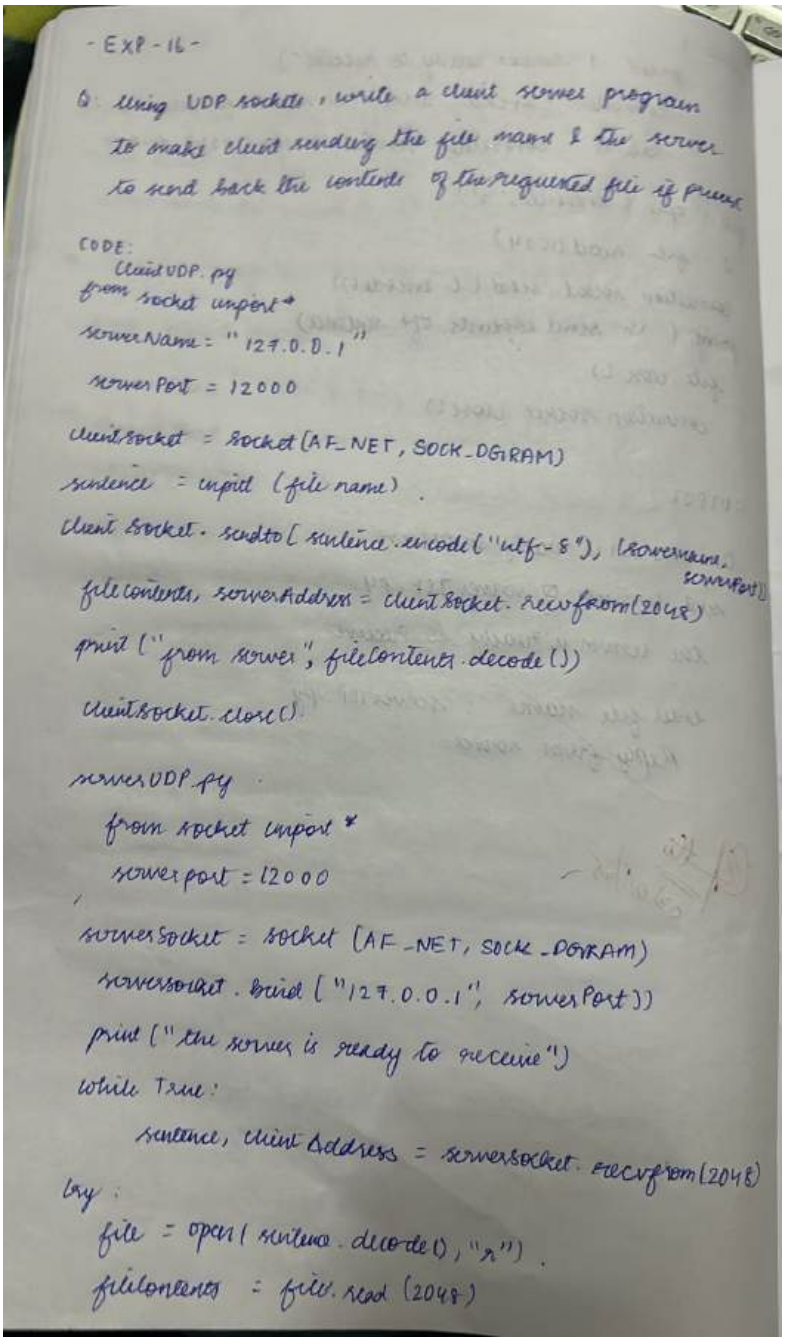
Output:



Program 4:

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.

Observation:



- EXP - 16 -

Q: Using UDP sockets, write a client server program to make client sending the file name & the server to send back the contents of the requested file if present

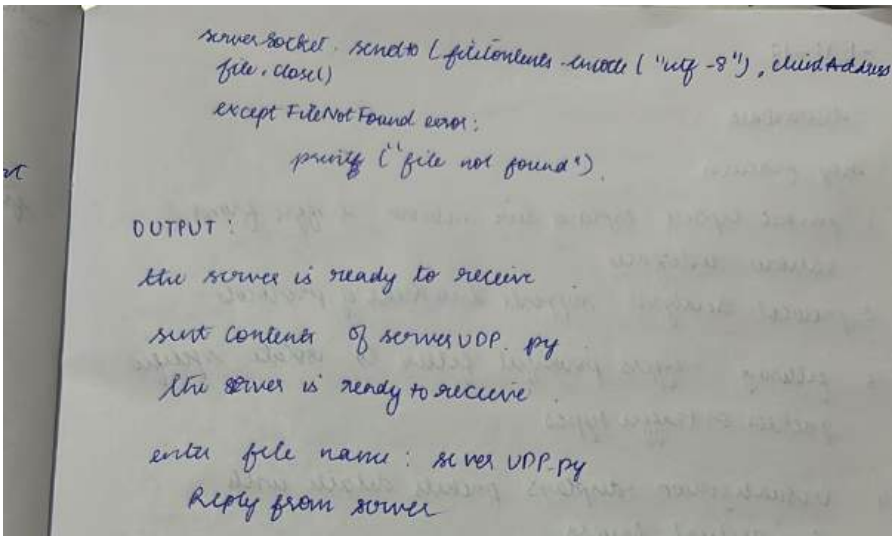
CODE:

ClientUDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_NET, SOCK_DGRAM)
sentence = input(file name)
clientSocket.sendto(sentence.encode("utf-8"), (servername, serverport))
fileContents, serverAddress = clientSocket.recvfrom(2048)
print("from server", fileContents.decode())
clientSocket.close()
```

serverUDP.py

```
from socket import *
serverport = 12000
serverSocket = socket(AF_NET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print("the server is ready to receive")
while True:
    sentence, clientAddress = serverSocket.recvfrom(2048)
try:
    file = open(sentence.decode(), "r")
    fileContents = file.read(2048)
```



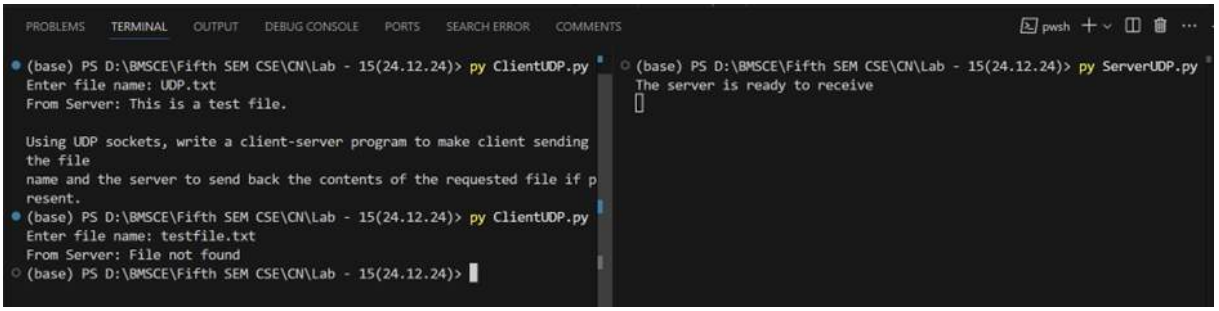
Serverudp.py

```
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()
```

Clienttudp.py

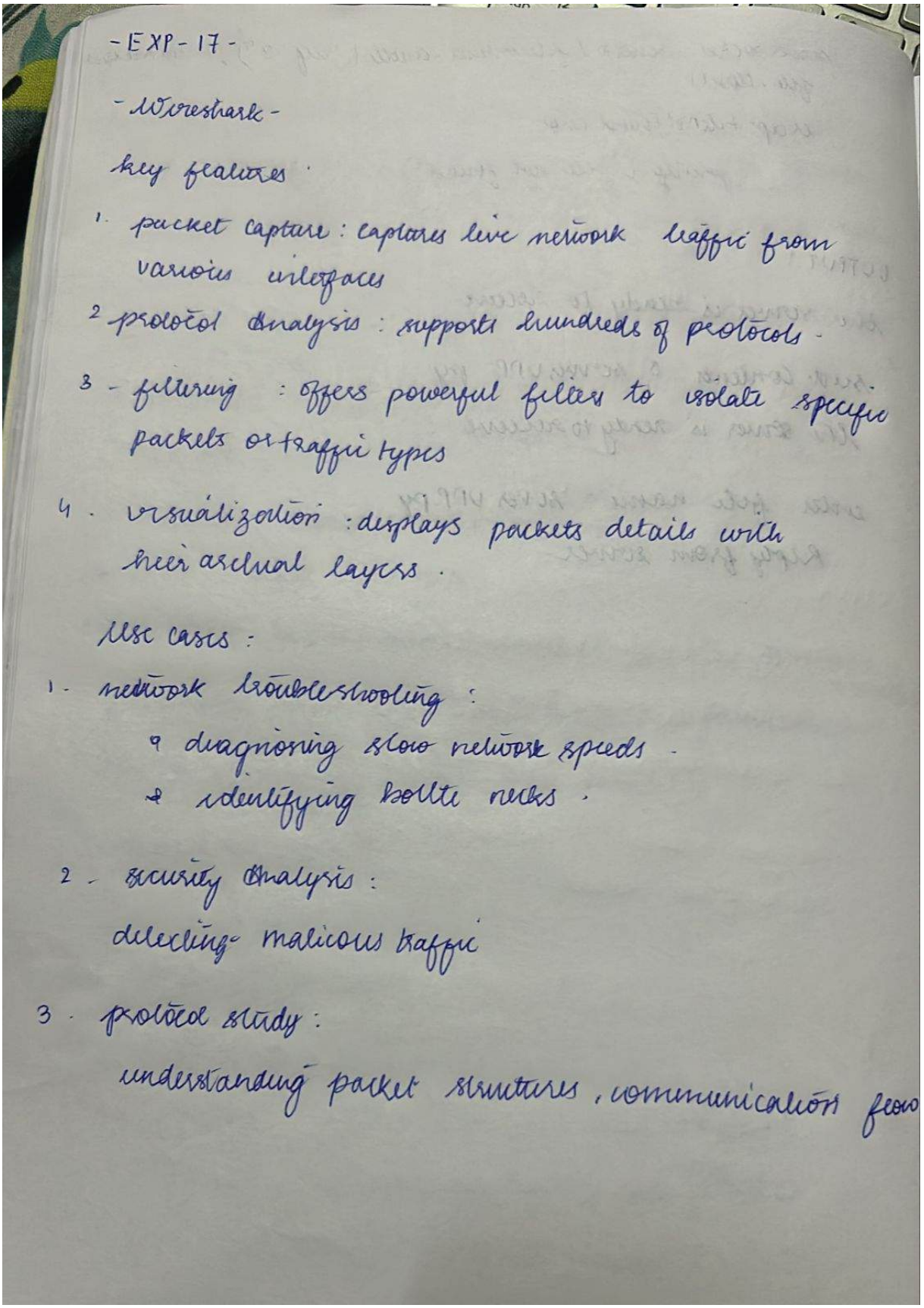
```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)

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()
```

Output:


PROBLEMS
TERMINAL
OUTPUT
DEBUG CONSOLE
PORTS
SEARCH ERROR
COMMENTS
pwsh
(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)> py ClientUDP.py
Enter file name: UDP.txt
From Server: This is a test file.
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 p
resent.
(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)> py ClientUDP.py
Enter file name: testfile.txt
From Server: File not found
(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)>
(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)> py ServerUDP.py
The server is ready to receive

Program 5:

Tool Exploration –Wireshark



- EXP - 17 -

- Wireshark -

key features:

1. packet capture: captures live network traffic from various interfaces
2. protocol analysis: supports hundreds of protocols.
3. filtering: offers powerful filters to isolate specific packets or traffic types
4. visualization: displays packets details with their actual layers.

use cases:

1. network troubleshooting:
   - diagnosing slow network speeds.
   - identifying bottle necks.
2. security analysis:
   detecting malicous traffic
3. protocol study:
   understanding packet structures, communication flow