VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

COMPUTER NETWORKS

Submitted by

GAYATRI VIKAS YATAGIRI(1BM22CS102)

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 **GAYATRI VIKAS YATAGIRI(1BM22CS102),** 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.

Dr. Latha N.R.
Associate Professor,
Department of CSE,
BMSCE, Bengaluru

Dr. Kavitha Sooda
Professor and Head,
Department of CSE
BMSCE, Bengaluru

INDEX

CYCLE 1

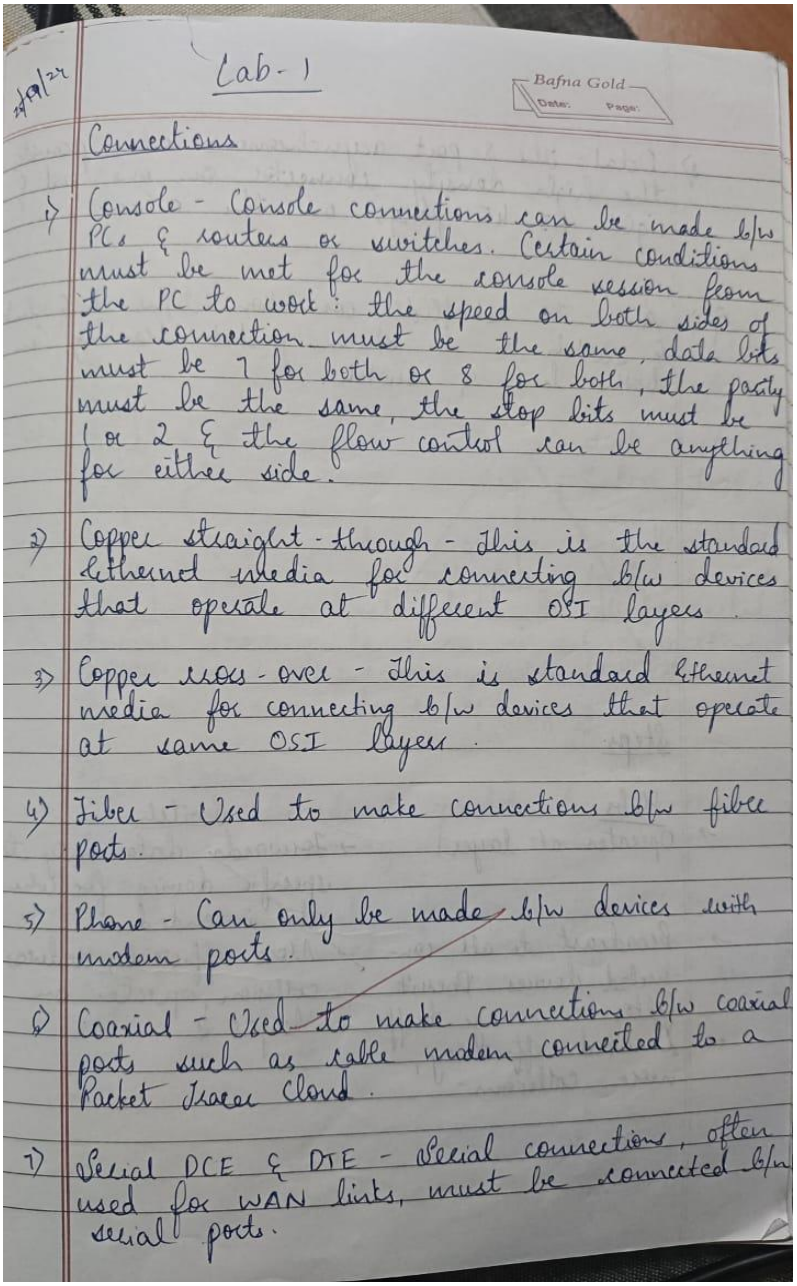
INDEX

CYCLE 2

CYCLE-1

PROGRAM1: 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



Lab-1

Bafna Gold

Connections

1) Console - Console connections can be made b/w PCs & routers or switches. Certain conditions must be met for the console session from the PC to work: the speed on both sides of the connection must be the same, data bits must be 7 for both or 8 for both, the parity must be the same, the stop bits must be 1 or 2 & the flow control can be anything for either side.

2) Copper straight-through - This is the standard Ethernet media for connecting b/w devices that operate at different OSI layers.

3) Copper cross-over - This is standard Ethernet media for connecting b/w devices that operate at same OSI layers.

4) Fiber - Used to make connections b/w fibre ports.

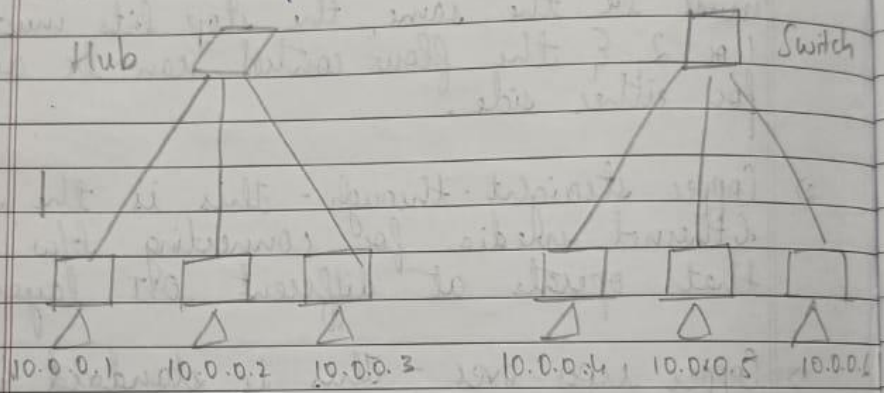
5) Phone - Can only be made b/w devices with modem ports.

6) Coaxial - Used to make connections b/w coaxial ports such as cable modem connected to a Packet Tracer Cloud.

7) Serial DCE & DTE - Serial connections, often used for WAN links, must be connected b/w serial ports.

8) Octal - The 8-port asynchronous cable provides the high density connector on one end & eight RJ-45 plugs on the other.

Q) Aim: To establish connection b/w PC & hub & switch & to show packet tracing in both hub & switch.



Steps

Hubs	Switch
→ Operates at layer 1	→ Forwards data only to specific device for which its intended
→ Broadcast to all connected devices. Doesn't filter or manage traffic.	→ More efficient, reduces collision operates on layer 2
→ Limited efficiency, more collisions.	

Bafna Gold
Date: Page:

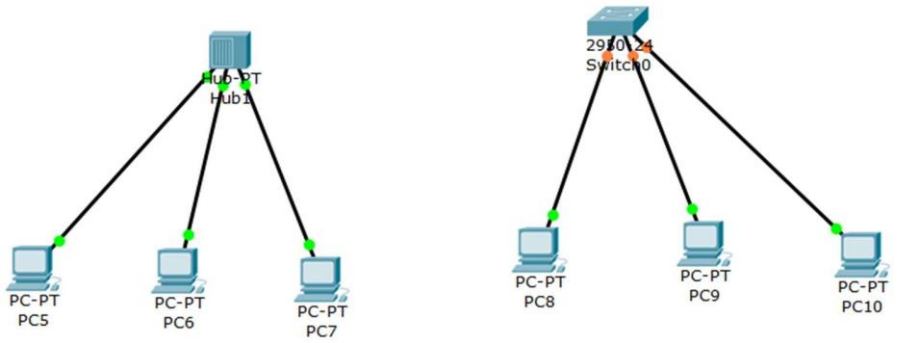
Procedure

→ Add hub 1, switch 1 & 6 PCs for the hub to the cisco packet tracer.
→ Use copper straight-through cables to connect PCs.
→ Assign IP addresses to each PC & subnet mask
→ Switch to simulations mode to observe data traffic behaviors when packets are sent b/w devices.
→ In the hub network, notice how all hub packets to all devices causing potential traffic overload.
→ In the switch network, observe how the switch forwards packets all the intended receipts
→ The hub broadcasts data to all connected devices leading to more network congestion, while the switch efficiently sends data only to the correct device, optimizing performance

Observations

→ The hub broadcasts packets to all the devices which may cause unnecessary traffic.
→ The switch only forwards packet to appropriate devices by learning MAC addresses, making it more efficient in reducing traffic.

TOPOLOGY:



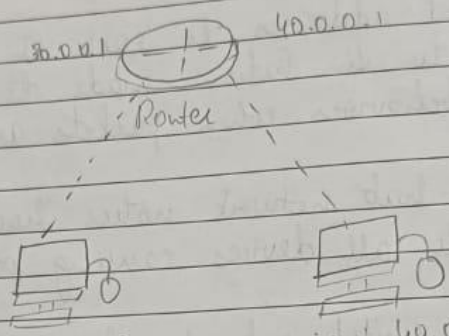
OUTPUT:
Pcs are connected

PROGRAM2: Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply
OBSERVATION:

9/10/24

Lab - 2

Aim: Configure IP addresses to routers in packet tracer. Explore the following messages ping responses, Destination unreachable request timed out, reply.



Topology
Two PCs (PC1 & PC2) are connected to router
Star topology

Procedure
→ Add a router, two PCs for star topology.
→ Use copper cross-over cables to connect PCs with routers.
→ Configure both PCs with IP addresses, gateway & net.
→ In CLI of Router
Enable
Config terminal
Interface fastethernet 0/0
ip address 30.0.0.1 255.0.0.0
no shutdown
exit
&

Bafna Gold
Date: Page:

interface fastethernet 1/0
ip address 40.0.0.1 255.0.0.0
no shutdown
exit

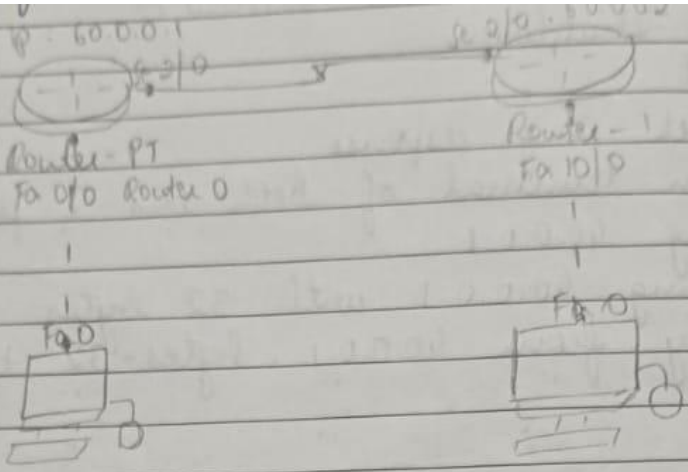
→ check ping response
Open terminal of both PCs & ping.
→ Ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data
Reply from 40.0.0.1 . bytes=32 time=0ms TTL=25
" " " " " "
" " " " " "
" " " " " "

Ping statistics
Packets : Sent = 4 Received = 4 Lost = 0 (0% loss)

Observations
→ In case of wrong ethernet configuration, requests will be timed out.
→ A successful ping means routing is set up correctly.

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
* - candidate default, U - per user static route, O - ODR
P - periodic downloaded static route

C 30.0.0.0/8 is directly connected FastEthernet 0/0
C 40.0.0.0/8 is directly connected, FastEthernet 1/0



ip address: 10.0.0.1
gateway: 10.0.0.3

ip address: 20.0.0.1
gateway: 20.0.0.5

Topology
Two PCs are connected to 2 seperate routers through copper wires.
The routers are connected by serial wiring (DCE)

Procedure
→ Added 2 PCs and a router for each PC.
→ Use copper cross-over cables to connect PC with router
→ Configure both PCs with ip address & gateway
→ Use Serial PCI to connect both routers
→ In CLI of router 0
Enable
Config terminal
Interface fastethernet 0/0
ip address 10.0.0.3 255.0.0.0
no shutdown
exit

Bafna Gold
Date: Page:

interface serial 2/0
ip address 60.0.0.1 255.0.0.0
no shutdown
exit

Observations
→ In CLI of router 1
Enable
Config terminal
interface fastethernet 0/0
ip address 20.0.0.1 255.0.0.0
no shutdown
exit

interface serial 2/0
ip address 60.0.0.2 255.0.0.0
no shutdown
exit

Observations
• Each router is connected to 1 node through copper cross-over & to other router by serial DCE. Even then both nodes cannot communicate with each other.

→ Check ping response from ~~router~~ PC 0.
ping 20.0.0.1
Pinging 20.0.0.01 with 32 bytes of date
Reply from 10.0.0.3: Destination host unreachable
" " "
" " "
Request timed out

Ping statistics for 20.0.0.1
Packets: Sent = 4 Recieved = 0 Lost = 4 (100% loss)

Ping 10.0.0.3 from 10.0.0.1
Pinging 10.0.0.3 with 32 bytes of data
Reply from 10.0.0.3: bytes=32 time=0ms TTL=255
" " " " "
" " " " "
" " " " "

Ping statistics for 10.0.0.3;
Packets: Sent = 4, Received = 4 Lost = 0 (0% loss)
Approx round trip times in ms:
min = 0ms, max = 0ms, avg = 0ms

Observations
Each router is connected to node & to other routes. But one node cannot communicate with other node as it is unaware of the ~~connection~~ network b/w both the routers.

Show ip route
C 20.0.0.0/8 is directly connected, FastEthernet 0/0
C 60.0.0.0/8 is directly connected, Serial 2/0

In CLI of router 0
ip route 10.0.0.0 255.0.0.0 60.0.0.1

In CLI of router 1
ip route 20.0.0.0 255.00.0 60.0.0.2

Ping 10.0.0.1 from 20.0.0.1
Pinging 10.0.0.1 with 32 bytes of data

Bafna Gold
Date: Page:

Reply from 10.0.0.1 bytes = 32 time = 0ms TTL = 255
" " " "
" " " "

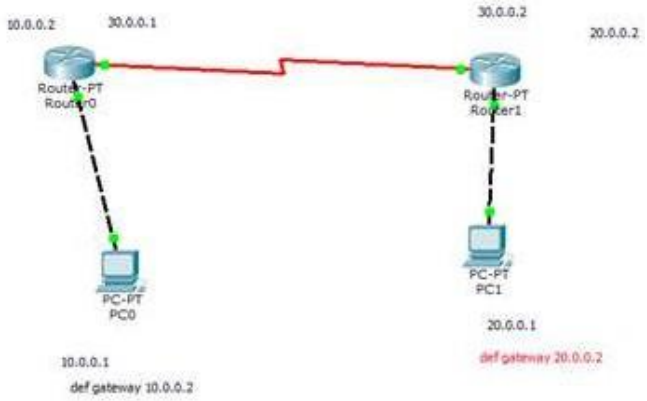
Ping statistics for 10.0.0.1
Packets: Sent = 4 Received = 4 Lost = 0 (0% loss)

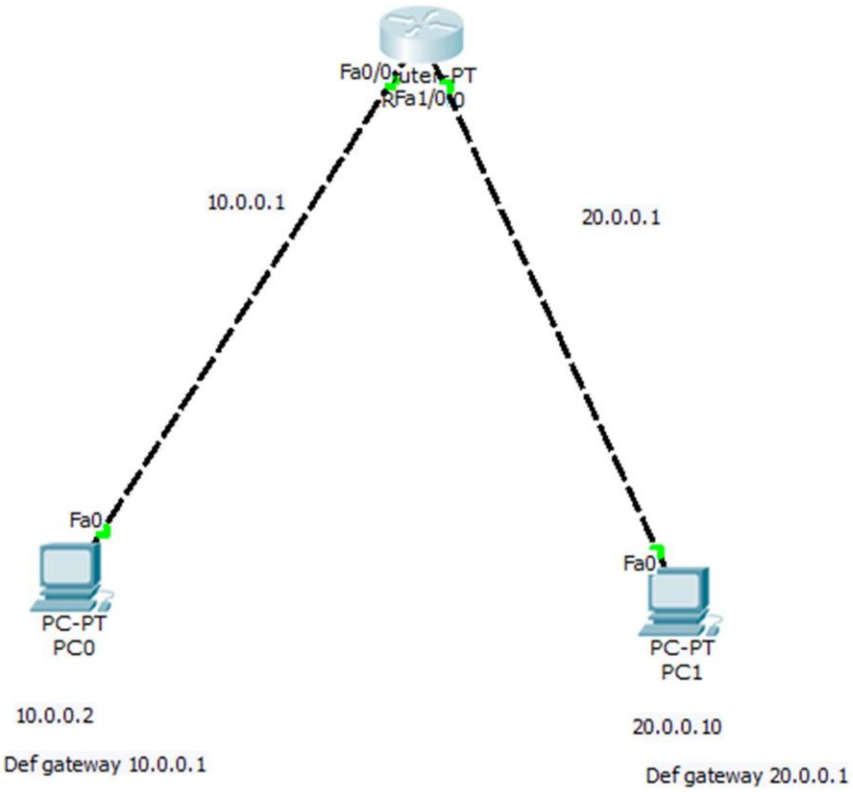
Observations
Node of 1 router is able to send packets to node of another router.

show ip route
S 10.0.0.0/8 [1/0] via 60.0.0.1
C 20.0.0.0/8 directly connected, FastEthernet 0/0
C 60.0.0.0/8 " " , Serial 2/0

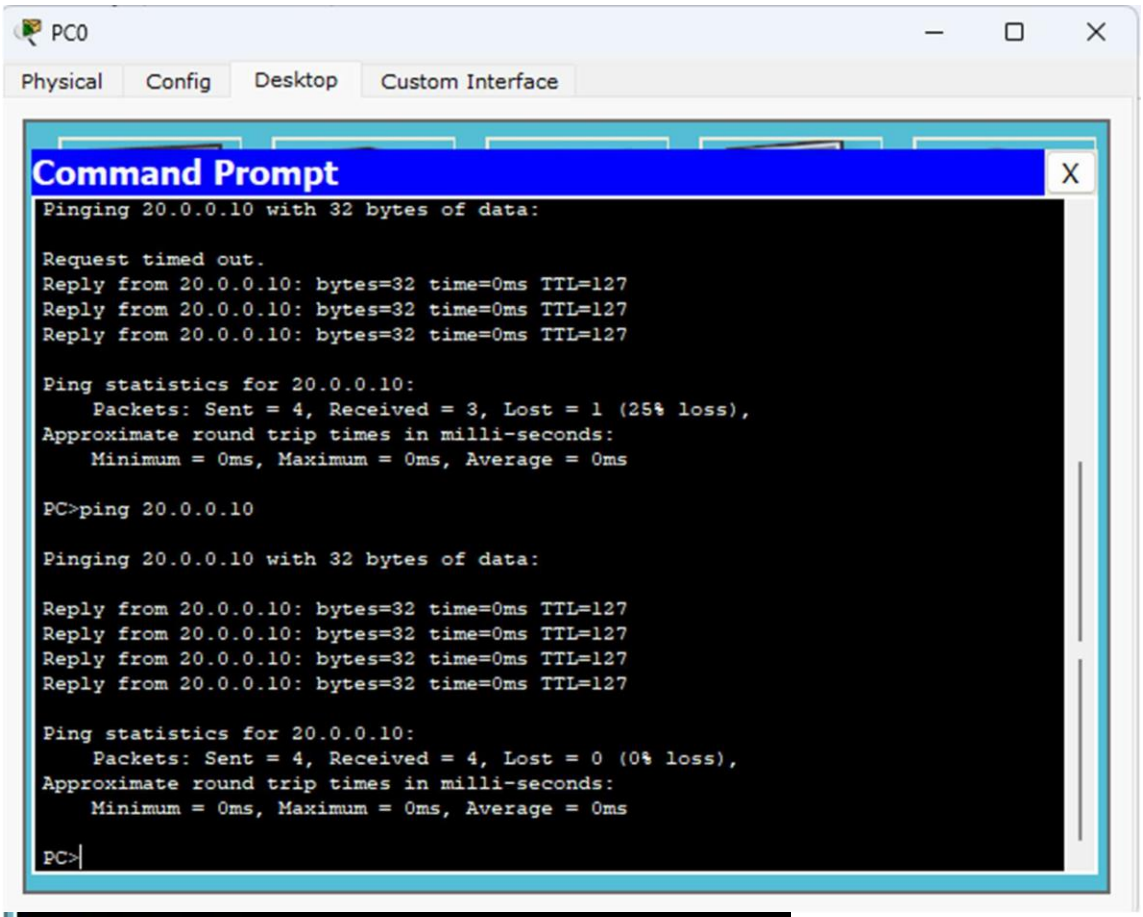
16/10/24

TOPOLOGY:

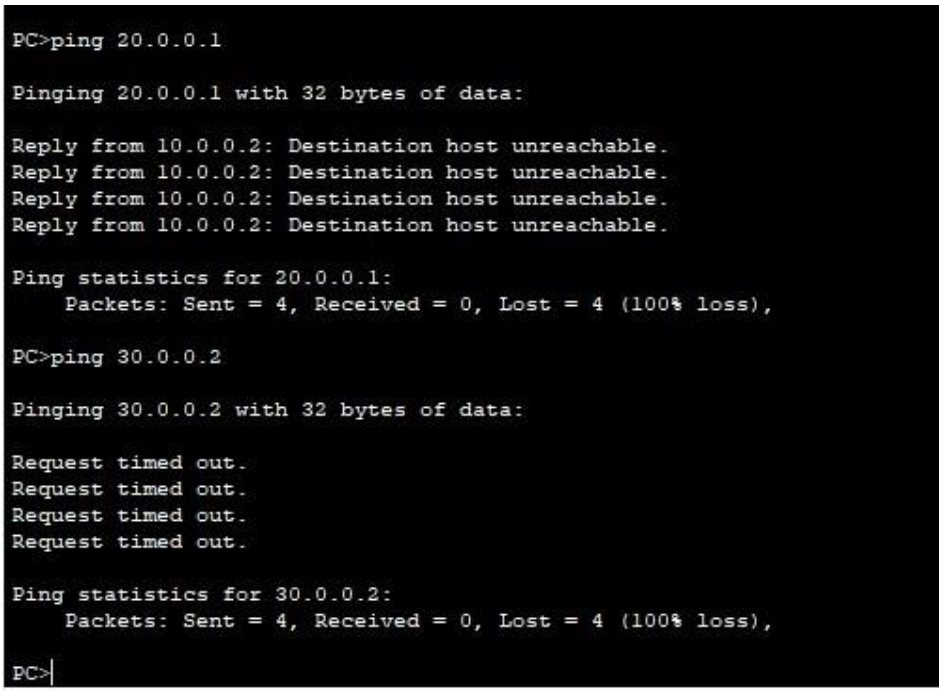

10.0.0.2
30.0.0.1
30.0.0.2
20.0.0.2
Router-PT
Router0
Router-PT
Router1
PC-PT
PC0
PC-PT
PC1
20.0.0.1
def gateway 20.0.0.2
10.0.0.1
def gateway 10.0.0.2


Fa0/0
Router-PT
Fa1/0/0
10.0.0.1
20.0.0.1
Fa0
Fa0
PC-PT
PC0
PC-PT
PC1
10.0.0.2
Def gateway 10.0.0.1
20.0.0.10
Def gateway 20.0.0.1

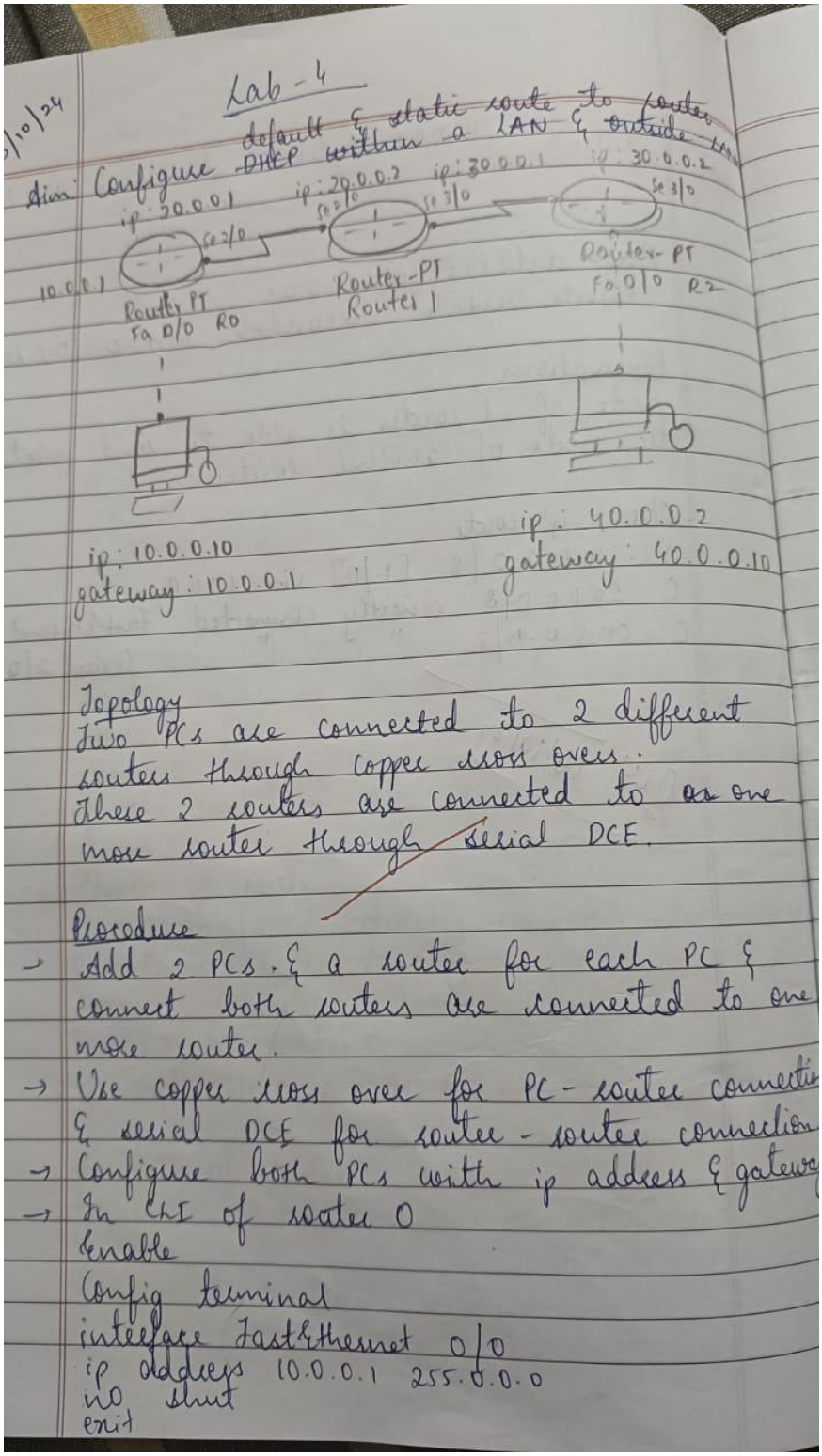
OUTPUT:




```
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=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data:

Reply from 30.0.0.1: bytes=32 time=0ms TTL=255
Reply from 30.0.0.1: bytes=32 time=0ms TTL=255
Reply from 30.0.0.1: bytes=32 time=0ms TTL=255
Reply from 30.0.0.1: bytes=32 time=0ms TTL=255

Ping statistics for 30.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```


PC>ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Ping statistics for 20.0.0.1:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>ping 30.0.0.2
Pinging 30.0.0.2 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>

PROGRAM3: Configure default route, static route to the Router
OBSERVATION:



interface serial 2/0
ip address 20.0.0.1 255.0.0.0
no shut
exit

→ In CLI of router 1
enable
config t
interface serial 2/0
ip address 20.0.0.2 255.0.0.0
no shut
exit

interface serial 3/0
ip address 30.0.0.10 255.0.0.0
no shut
exit

→ In CLI of router 2
enable
Config t
Interface FastEthernet 0/0
ip address 40.0.0.10 255.0.0.0
no shut
exit

interface serial 3/0
ip address 30.0.0.20 255.0.0.0
no shut
exit

Now in CLI of router 0
enable
config t
ip route 0.0.0.0 0.0.0.0 20.0.0.2
exit

In CLI of router 2
enable
config t
ip route 0.0.0.0 0.0.0.0 30.0.0.1
exit

Open terminal of PC0
ping 40.0.0.2
Pinging 40.0.0.2 with 32 bytes of data
Reply from 40.0.0.2 : bytes=32 time=5ms TTL=127
" " " " " time=7ms "
Ping statistics for 40.0.0.2
Packets Sent=4 Received=3 Lost=1 (25% loss)
Approx round trip time in ms
min=5ms max=7ms avg=5ms

Open terminal of PC1
ping 10.0.0.10
Pinging 10.0.0.10 with 32 bytes of data
Reply from 10.0.0.10 : bytes=32 time=7ms TTL=125
" " " time=6ms "
" " " time=6ms "
" " " time=18ms "
Ping statistics for 10.0.0.10
Packets Sent=4 Received=4 Lost=0 (0% loss)

Bafna Gold
Date: Page:

Approx round trip time in ms
Min = 6ms Max = 18 ms Avg = 9ms

Observations
Each PC is now connected to all the 3 routers & other PC. So packets can be sent

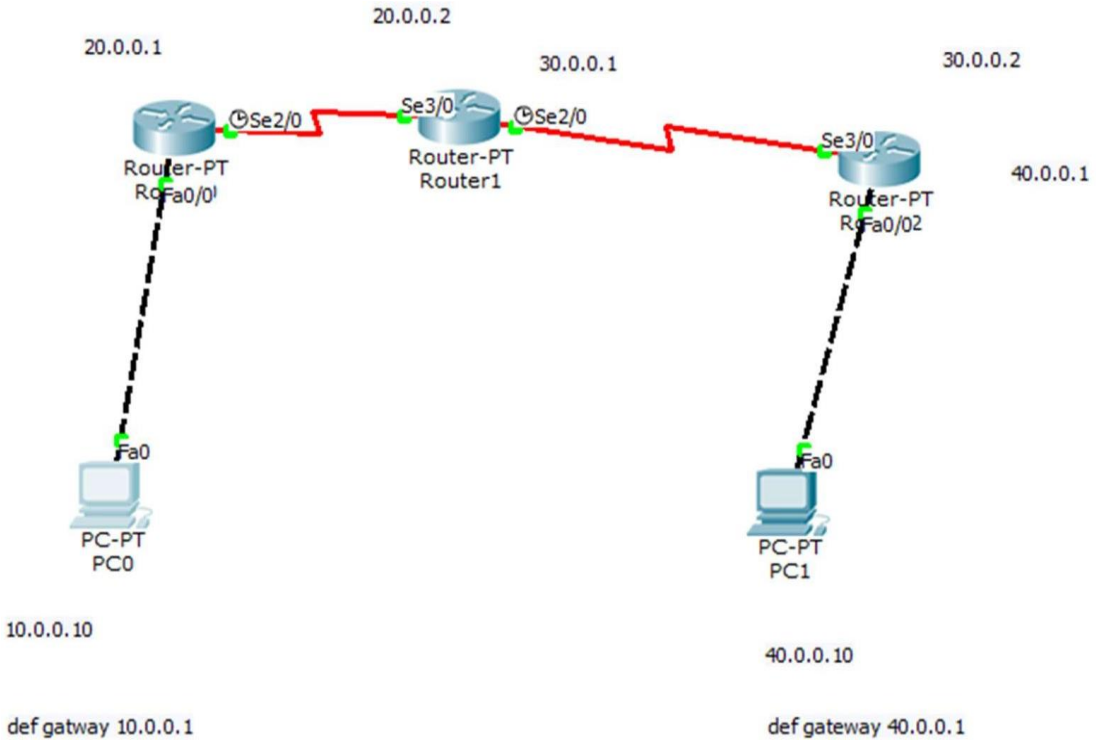
Show ip route (in CLI of router 0)
C 10.0.0.0/8 is directly connected, FastEthernet0
C 20.0.0.0/8 is directly connected Serial 2/0
S* 0.0.0.0/0 [1/0] via 20.0.0.2

In CLI of router 2
Show ip route
C 30.0.0.0/8 is directly connected Serial 3/0
C 40.0.0.0/8 is directly connected FastEthernet 0/
S* 0.0.0.0/0 [1/0] via 30.0.0.1

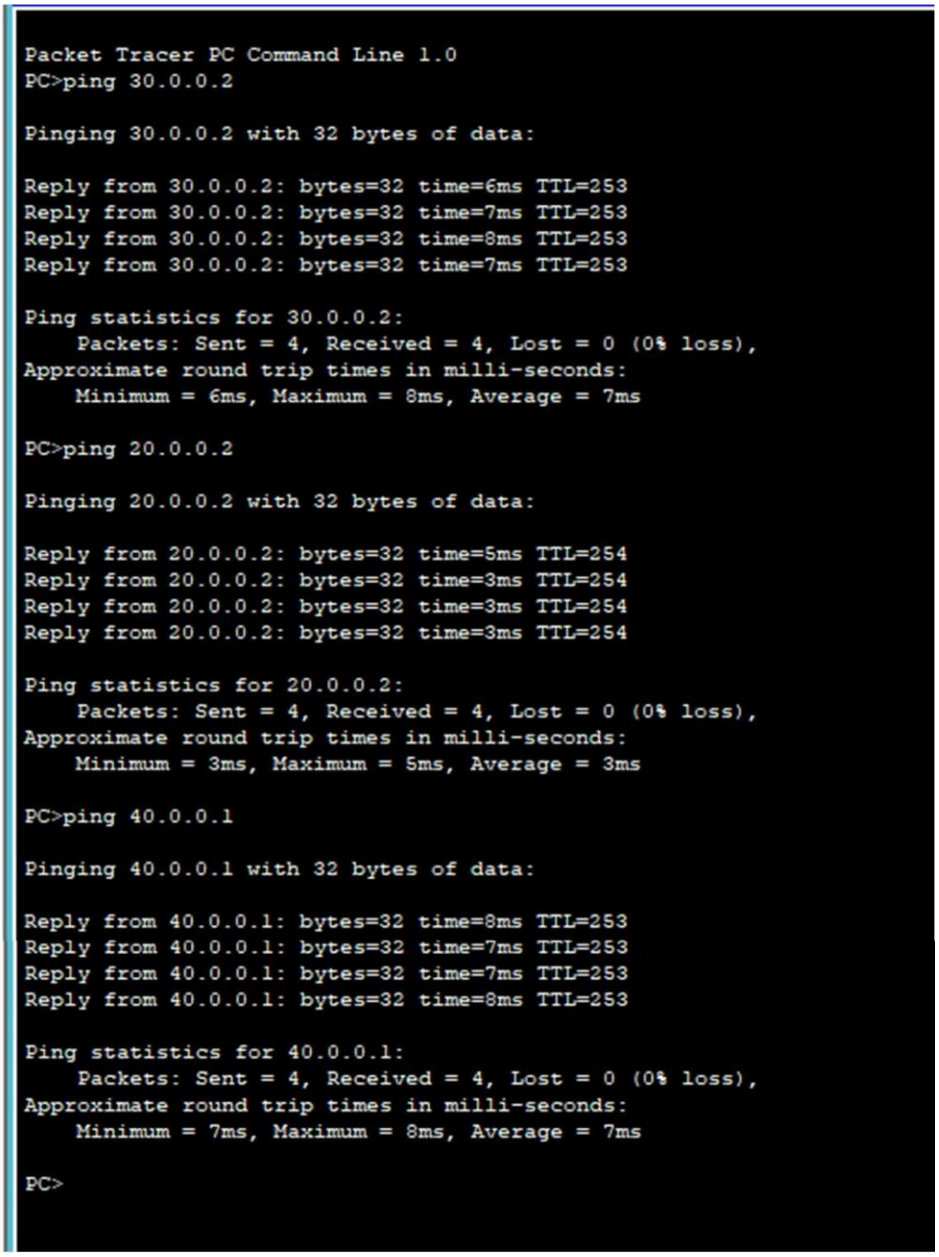
In CLI of router 1
show ip route
S 10.0.0.0/8 [1/0] via 20.0.0.1
C 20.0.0.0/8 is directly connected, Serial 2/0
C 30.0.0.0/8 is directly connected, Serial 3/0
S 40.0.0.0/8 [1/0] via 30.0.0.2

Lee
23/10/24

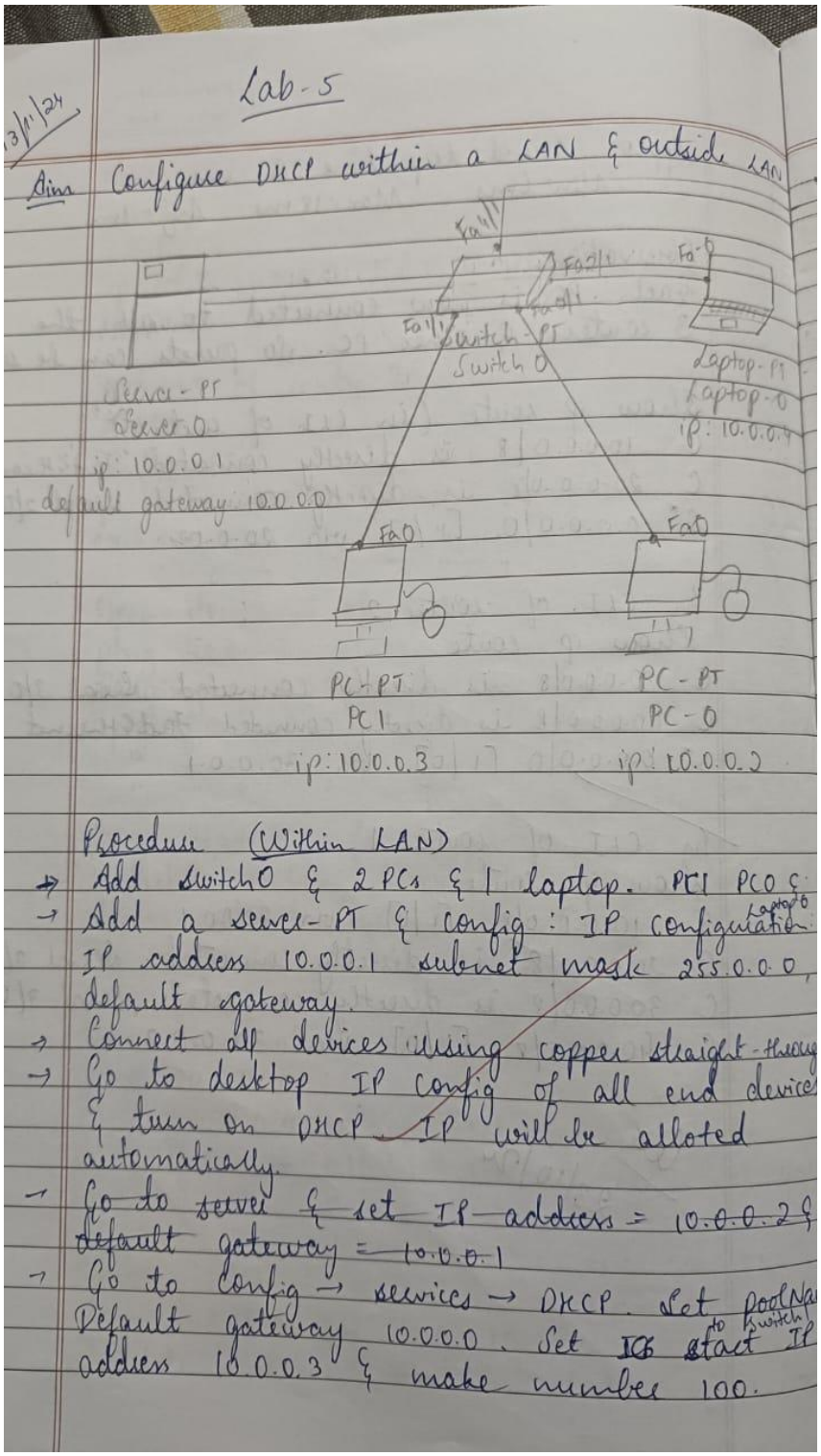
TOPOLOGY:

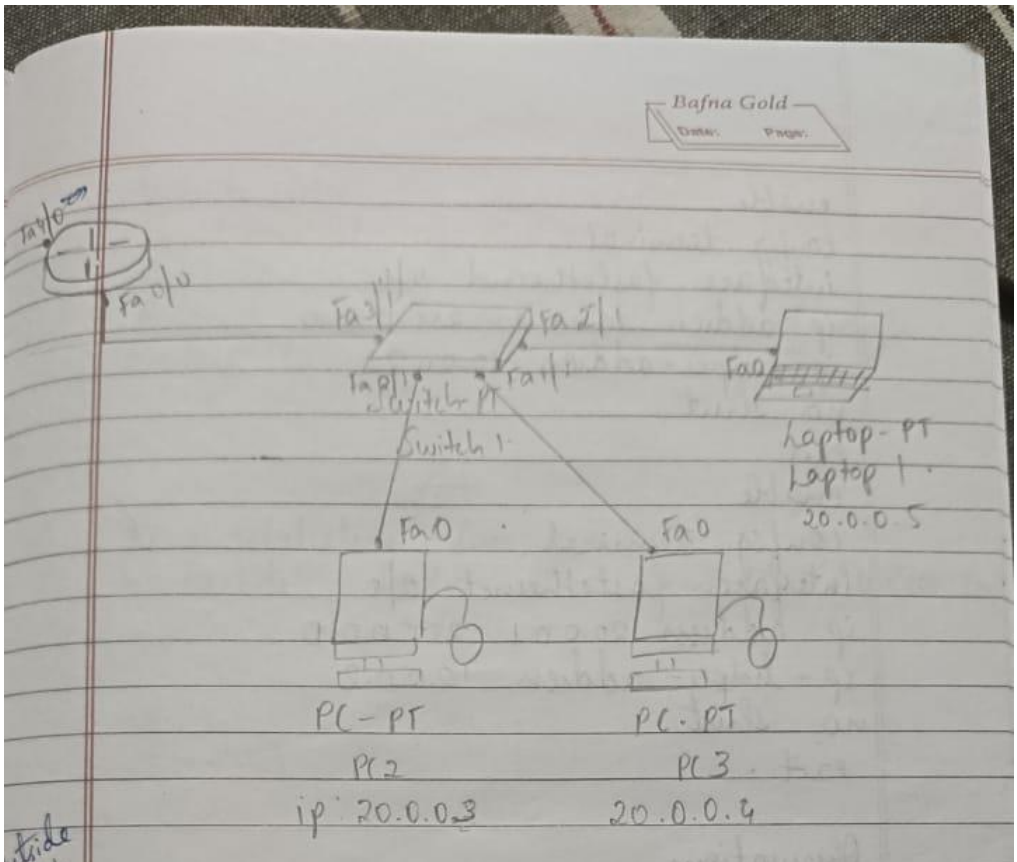

20.0.0.2
20.0.0.1
30.0.0.1
30.0.0.2
Se2/0
Se3/0
Se2/0
Se3/0
Router-PT
Router1
Router-PT
Router-PT
40.0.0.1
Fa0/01
Fa0/02
Fa0
Fa0
PC-PT
PC0
PC-PT
PC1
10.0.0.10
40.0.0.10
def gatway 10.0.0.1
def gateway 40.0.0.1

OUTPUT:


Packet Tracer PC Command Line 1.0
PC>ping 30.0.0.2
Pinging 30.0.0.2 with 32 bytes of data:
Reply from 30.0.0.2: bytes=32 time=6ms TTL=253
Reply from 30.0.0.2: bytes=32 time=7ms TTL=253
Reply from 30.0.0.2: bytes=32 time=8ms TTL=253
Reply from 30.0.0.2: bytes=32 time=7ms TTL=253
Ping statistics for 30.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 6ms, Maximum = 8ms, Average = 7ms
PC>ping 20.0.0.2
Pinging 20.0.0.2 with 32 bytes of data:
Reply from 20.0.0.2: bytes=32 time=5ms TTL=254
Reply from 20.0.0.2: bytes=32 time=3ms TTL=254
Reply from 20.0.0.2: bytes=32 time=3ms TTL=254
Reply from 20.0.0.2: bytes=32 time=3ms TTL=254
Ping statistics for 20.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 3ms, Maximum = 5ms, Average = 3ms
PC>ping 40.0.0.1
Pinging 40.0.0.1 with 32 bytes of data:
Reply from 40.0.0.1: bytes=32 time=8ms TTL=253
Reply from 40.0.0.1: bytes=32 time=7ms TTL=253
Reply from 40.0.0.1: bytes=32 time=7ms TTL=253
Reply from 40.0.0.1: bytes=32 time=8ms TTL=253
Ping statistics for 40.0.0.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 7ms, Maximum = 8ms, Average = 7ms
PC>

PROGRAM4: Configure DHCP within a LAN and outside LAN.
OBSERVATION:





outside LAN

→ Add another switch & connect to it PC2, PC3 & Laptop-1

→ Also add a router to which connect both the switches through 'Automatically choose connection type'.

→ Now go to server IP config & change IP address 10.0.0.2 & default gateway 10.0.0.1 Change the pool name to switch1 → default gateway 10.0.0.1 → Save

→ Create another poolName switch2 & set default gateway 20.0.0.1 → start IP address 20.0.0.3 → max number of users: 100 → Add

→ Go to router terminal

→ In CLI of the router, type the following commands:

enable
config terminal
interface fastethernet 4/0
ip address 10.0.0.1 255.0.0.0
ip helper-address 10.0.0.2
no shut
exit
enable
config terminal
interface fastethernet 0/0
ip address 20.0.0.1 255.0.0.0
ip helper-address 10.0.0.2
no shut
exit.

Observations
Within LAN
In terminal of PC0
ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data
Reply from 10.0.0.3 : bytes=32 time=1ms TTL=12
" " " " time=0ms "
" " " " " "
" " " " " "

Ping statistics for 10.0.0.3 :
Packets: Sent=4 Received=4 Lost=0 (0% loss)
Approximate round trip time in ms:
min=0ms max=1ms avg=0ms

Bafna Gold
Date: Page:

Outside LAN

In terminal of PC0

ping 20.0.0.3

Pinging 20.0.0.3 with 32 bytes of data

Request timed out

" "

" "

" "

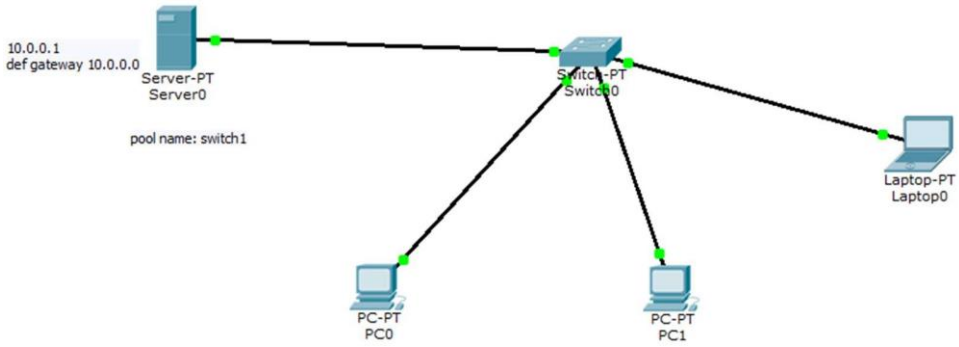
Ping statistics for 10.0.0.7:

Packets: Sent=4 Received=0 Lost=4 (100% loss)

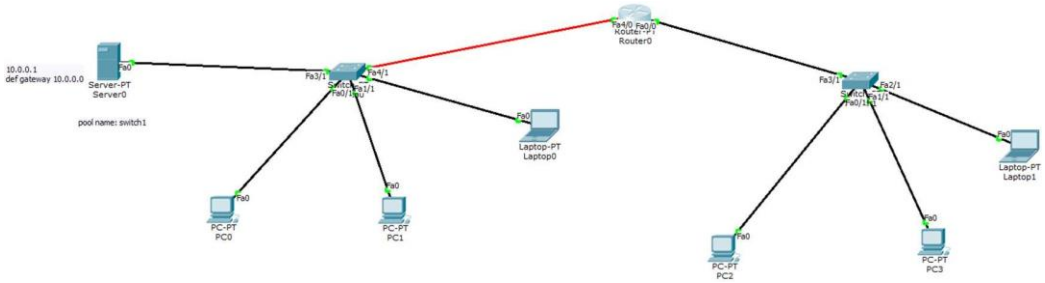
8th

13/11/24

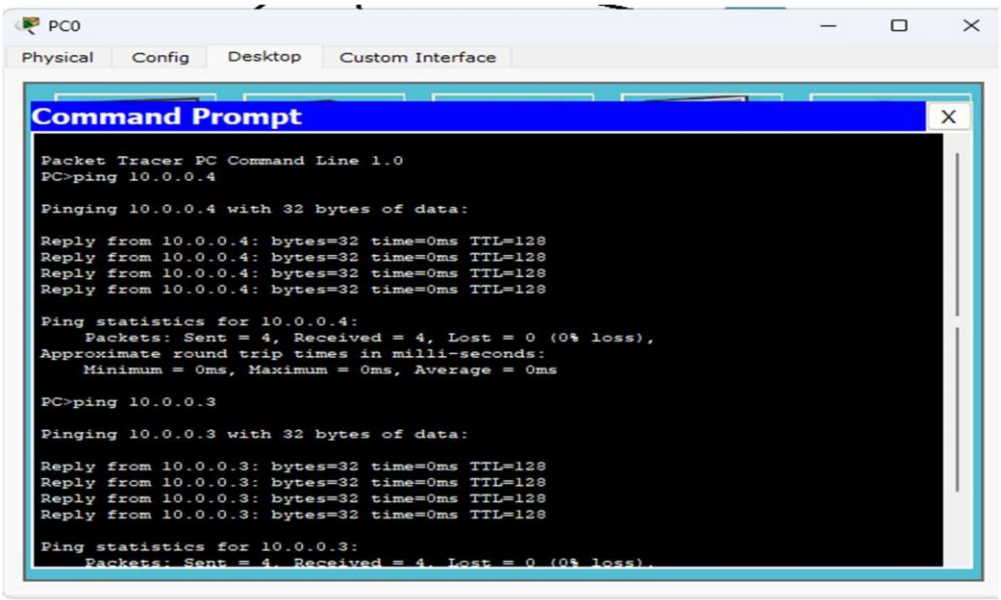
TOPOLOGY:
Within lan



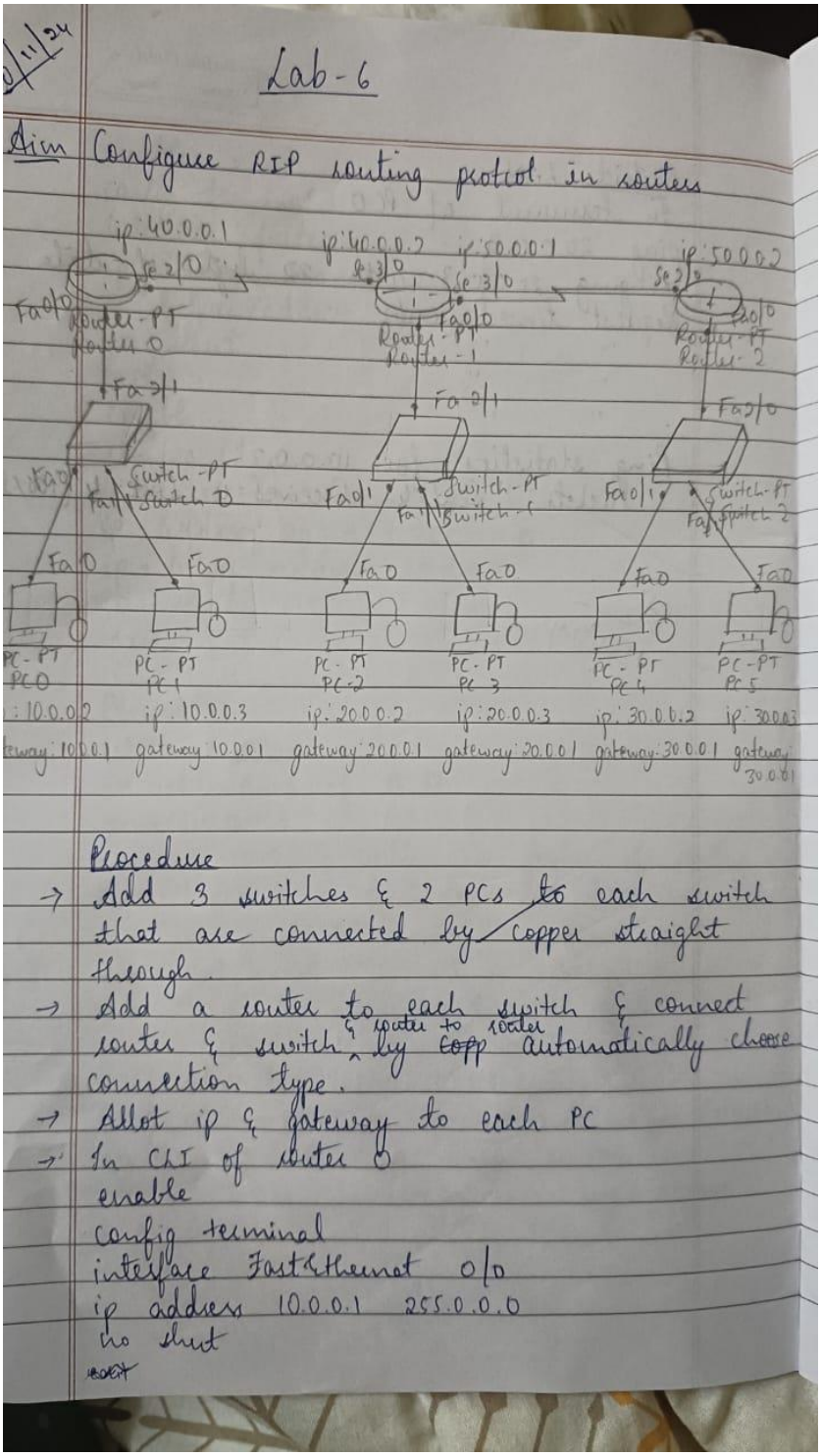
Outside lan



OUTPUT:



PROGRAM5: Configure RIP routing Protocol in Routers
OBSERVATION:



Bafna Gold
Date: Page:

interface serial 2/0
ip address 40.0.0.1 255.0.0.0
no shut
exit

→ In CLI of router 1
interface FastEthernet 0/0
ip address 20.0.0.1 255.0.0.0
no shut
exit
interface serial 2/0
ip address 40.0.0.2 255.0.0.0
no shut
exit
interface serial 3/0
ip address 50.0.0.1 255.0.0.0
no shut
exit

→ In CLI of router 2
interface FastEthernet 0/0
ip address 30.0.0.1 255.0.0.0
no shut
exit
interface serial 2/0
ip address 50.0.0.2 255.0.0.0
no shut
exit

All the routers are configured now.

→ In CLI of router 0:
enable
config t
router rip
network 10.0.0.0
network 40.0.0.0

→ In CLI of router 1
enable
config t
router rip
network 40.0.0.0
network 50.0.0.0
network 20.0.0.0

→ In CLI of router 2
enable
config t
router rip
network 50.0.0.0
network 30.0.0.0

Observations

In CLI of router 0
show ip route
C 10.0.0.0/8 is directly connected, FastEthernet 0/0
R 20.0.0.0/8 [120/1] via 40.0.0.2, 00:00:14, Serial 2/0
R 30.0.0.0/8 [120/2] " " " " "
C 40.0.0.0/8 is directly connected, Serial 2/0
R 50.0.0.0/8 [120/1] via 40.0.0.2, 00:00:14, Serial 2/0

In CLI of router 1
show ip route
R 10.0.0.0/8 [120/1] via 40.0.0.1, 00:00:22, Serial 2/0

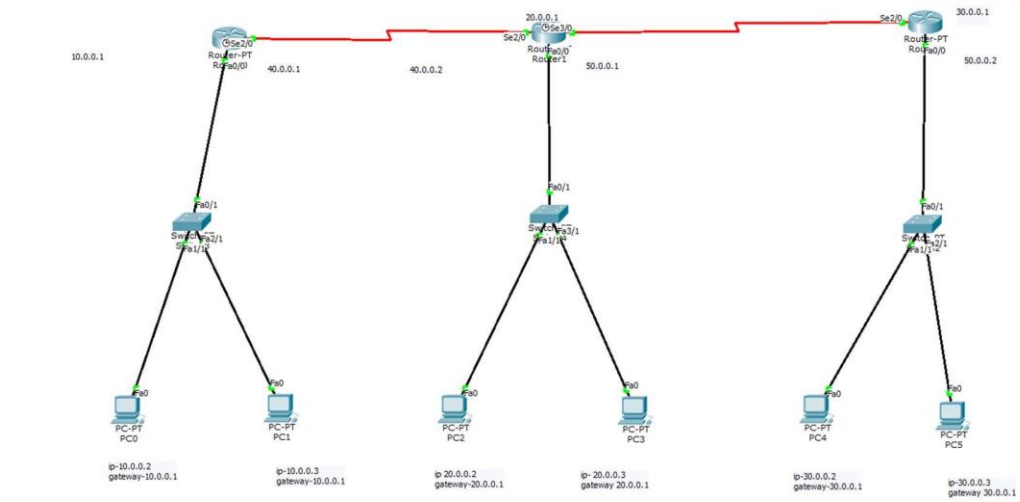
Bafna Gold
Date: Page:

C 20.0.0.0/8 is directly connected, FastEthernet 0/0
C 40.0.0.0/8 " " " , Serial 2/0
C 50.0.0.0/8 " " " , Serial 3/0

In CLI of router 2
show ip route
R 10.0.0.0/8 [120/2] via 50.0.0.1, 00:00:11, Serial 2/0
R 20.0.0.0/8 [120/1] " " " "
C 30.0.0.0/8 is directly connected, FastEthernet 0/0
R 40.0.0.0/8 [120/1] via 50.0.0.1, 00:00:11, Serial 2/0
C 50.0.0.0/8 is directly connected, Serial 2/0

In terminal of PC2
ping 30.0.0.3
Pinging 30.0.0.3 with 32 bytes of data:
Request timed out.
Reply from 30.0.0.3: bytes=32 time=4ms TTL=126
" " " " time " "
" " " " time=3ms "
Ping statistics for 30.0.0.3:
Packets: Sent=4 Received=3 Lost=1 (25% Loss)
Approximate round trip in ms:
min=3ms max=4ms avg=3ms

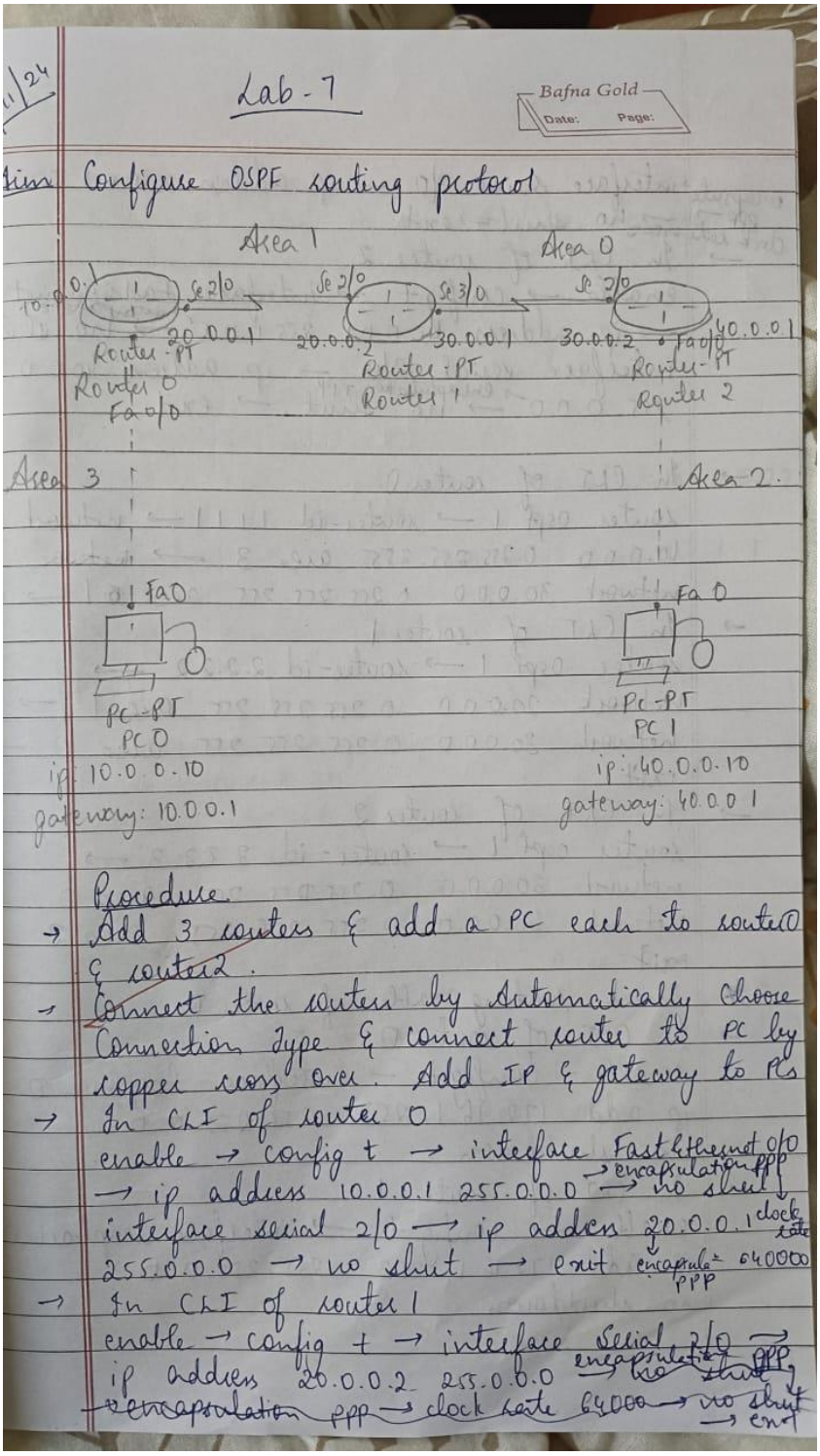
TOPOLOGY:



OUTPUT:

```
Request timed out.
Reply from 20.0.0.2: bytes=32 time=2ms TTL=126
Reply from 20.0.0.2: bytes=32 time=2ms TTL=126
Reply from 20.0.0.2: bytes=32 time=4ms TTL=126

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 4ms, Average = 2ms

PC>ping 20.0.0.2

Pinging 20.0.0.2 with 32 bytes of data:

Reply from 20.0.0.2: bytes=32 time=5ms TTL=126
Reply from 20.0.0.2: bytes=32 time=2ms TTL=126
Reply from 20.0.0.2: bytes=32 time=1ms TTL=126
Reply from 20.0.0.2: bytes=32 time=1ms TTL=126

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 5ms, Average = 2ms

PC>
```

PROGRAM6: Configure OSPF routing protocol
OBSERVATION:



encapsulation PPP → clock rate 64000 → interface serial 3/0 → ip address 30.0.0.1 255.0.0.0 → no shut → exit

→ In CLI of router 2
enable → config t → interface FastEthernet 0/0 → ip address 40.0.0.1 255.0.0.0 → no shut
interface serial 2/0 → ip address 40.0.0.2 255.0.0.0 → encapsulation PPP → no shut → exit.

→ In CLI of router 0
router ospf 1 → router-id 1.1.1.1 → network 10.0.0.0 0.255.255.255 area 3 → network 30.0.0.0 0.255.255.255 area 0 1 → exit

→ In CLI of router 1
router ospf 1 → router-id 2.2.2.2 → network 20.0.0.0 0.255.255.255 area 1 → network 30.0.0.0 0.255.255.255 area 0 → exit

→ In CLI of router 2
router ospf 1 → router-id 3.3.3.3 → network 30.0.0.0 0.255.255.255 area 0 network 40.0.0.0 0.255.255.255 area 2 exit

★ Check routing table of router 0

→ In CLI of router 0
enable - config t → interface loopback 0
ip add 172.16.1.252 255.255.0.0
no shutdown

→ In CLI of router 1
enable → config t → interface loopback 0
ip add 172.16.1.253 255.255.0.0
no shutdown

Bafna Gold
Date: Page:

→ In CLI of router 2
enable → config t → interface loopback 0
ip add 172.16.1.254 255.255.0.0 →
no shutdown

→ Check routing table of router 2.

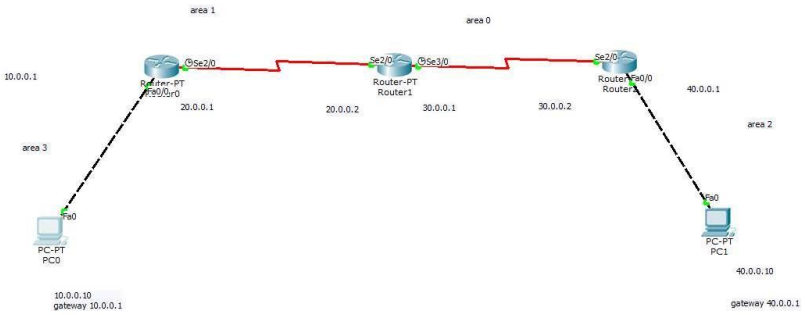
In CLI of router 0
router ospf 1 → area 1 virtual-link 2.2.2.2
→ In CLI of router 1
router ospf 1 → area 1 virtual-link 1.1.1.1

→ Check routing table of router 2.

Output
In command prompt of PC 0
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=128
" " " time=8ms "
" " " time=7ms "
Ping statistics for 40.0.0.10:
Packets Sent=4 Received=3 Lost=1 (25% loss)
Approx round trip in ms:
min=6ms max=8ms avg=7ms

Lab
27/11/24

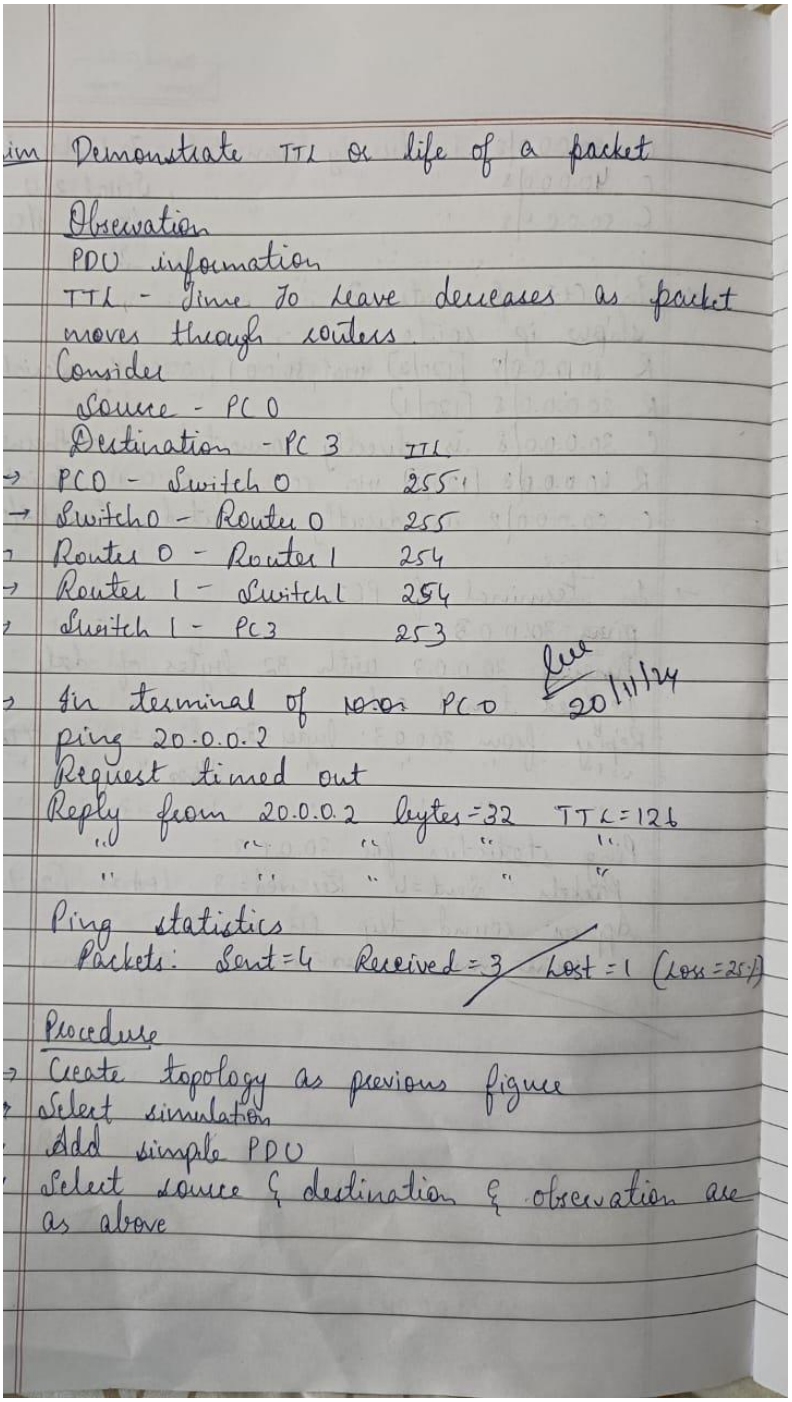
TOPOLOGY:



OUTPUT:

```
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=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=4ms 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 = 4ms, Maximum = 6ms, Average = 5ms

PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Reply from 40.0.0.10: bytes=32 time=8ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 8ms, Average = 6ms
```

PROGRAM7: Demonstrate the TTL/ Life of a Packet
OBSERVATION:



Aim Demonstrate TTL or life of a packet

Observation
PDU information
TTL - time to leave decreases as packet moves through routers.
Consider
Source - PC 0
Destination - PC 3

	TTL
PC0 - Switch 0	255
Switch0 - Router 0	255
Router 0 - Router 1	254
Router 1 - Switch 1	254
Switch 1 - PC3	253

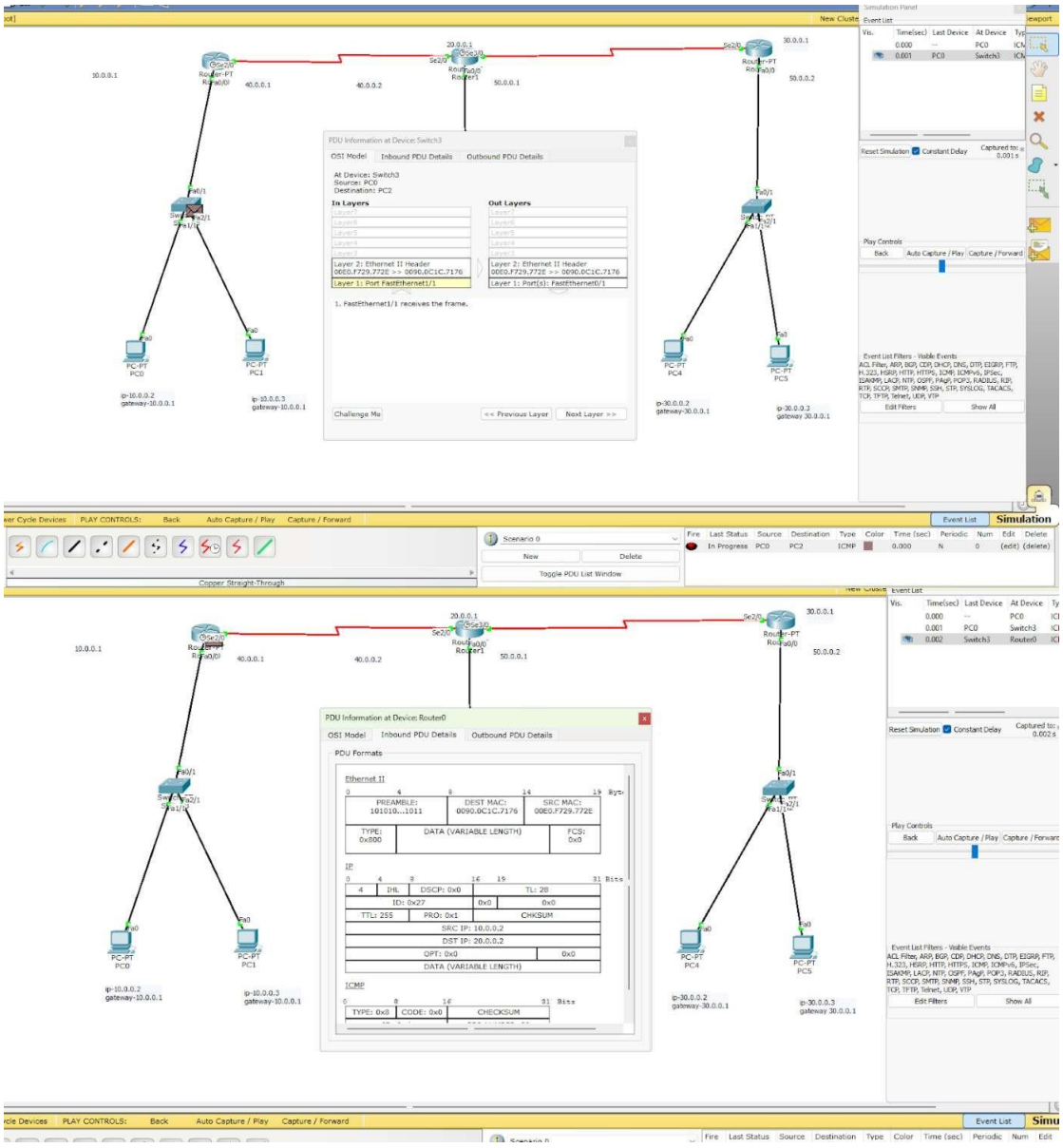
Lue 20/11/24

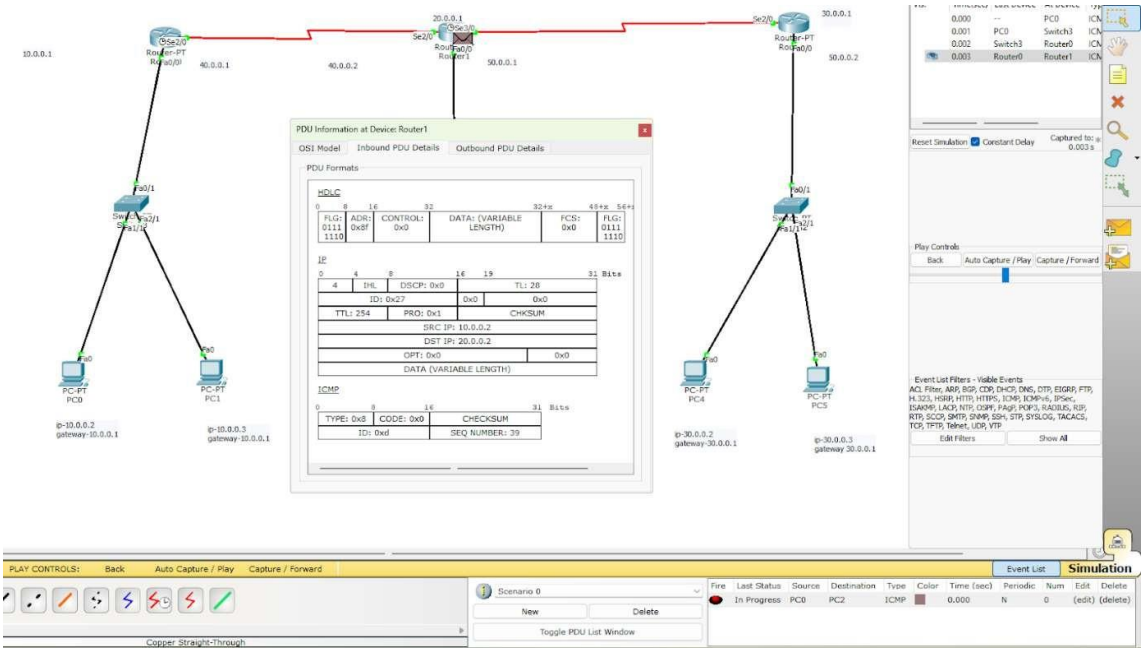
In terminal of PC 0
ping 20.0.0.2
Request timed out
Reply from 20.0.0.2 bytes=32 TTL=126
" " " " "
" " " " "

Ping statistics
Packets: Sent=4 Received=3 Lost=1 (Loss=25%)

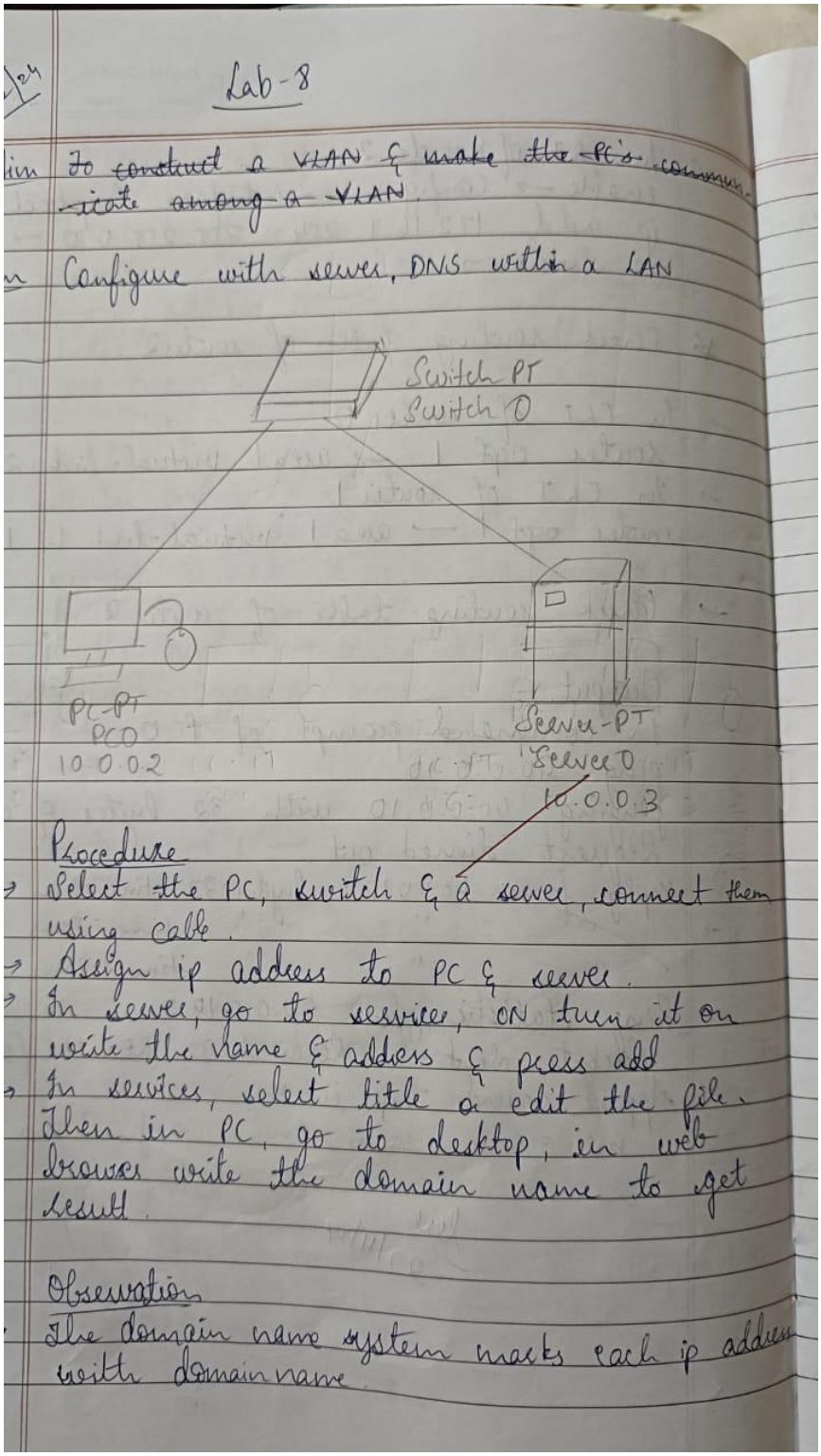
Procedure
Create topology as previous figure
Select simulation
Add simple PDU
Select source & destination & observation are as above

TOPOLOGY:

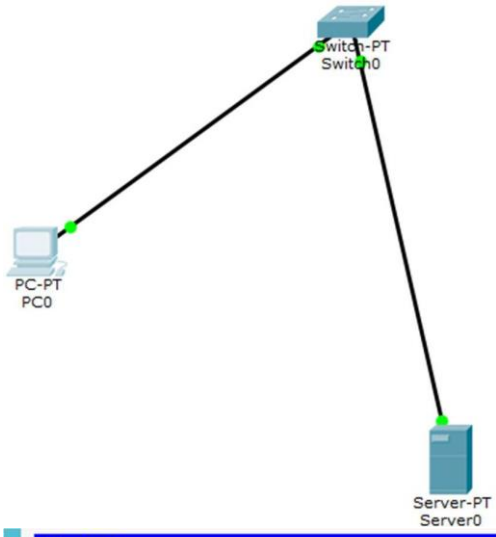

PDU Information at Device: Switch3
OSI Model
Inbound PDU Details
Outbound PDU Details
At Device: Switch3
Source: PC0
Destination: PC2
In Layers
Out Layers
Layer 2: Ethernet II Header 0060.F729.772E >> 0090.0C1C.7176
Layer 1: Port FastEthernet1/1
Layer 1: Port(s): FastEthernet0/1
1. FastEthernet1/1 receives the frame.
PDU Information at Device: Router0
PDU Formats
Ethernet II
PREAMBLE: 10101010...10101011
DEST MAC: 0090.0C1C.7176
SRC MAC: 0060.F729.772E
TYPE: 0x800
DATA (VARIABLE LENGTH)
FCS: 0x0
IP
SRC IP: 10.0.0.2
DST IP: 20.0.0.2
ICMP
TYPE: 0x0
CODE: 0x0
CHECKSUM
Simulation
Event List
Scenario 0
New
Delete
Toggle PDU List Window

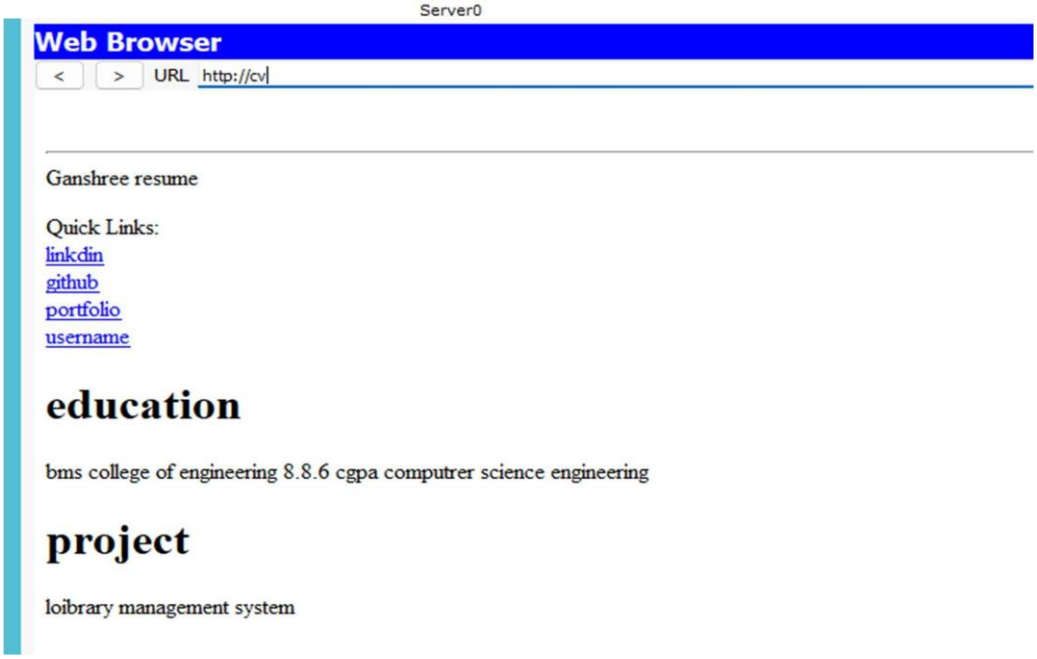

PDU Information at Device: Router1
OSI Model
Inbound PDU Details
Outbound PDU Details
PDU Formats
HDLC
IP
ICMP
PC-PT PC0
PC-PT PC1
PC-PT PC4
PC-PT PC5
PLAY CONTROLS:
Back
Auto Capture / Play
Capture / Forward
Event List
Simulation
Scenario 0
New
Delete
Toggle PDU List Window
Copper Straight-Through

PROGRAM8: Configure Web Server, DNS within a LAN.
OBSERVATION:

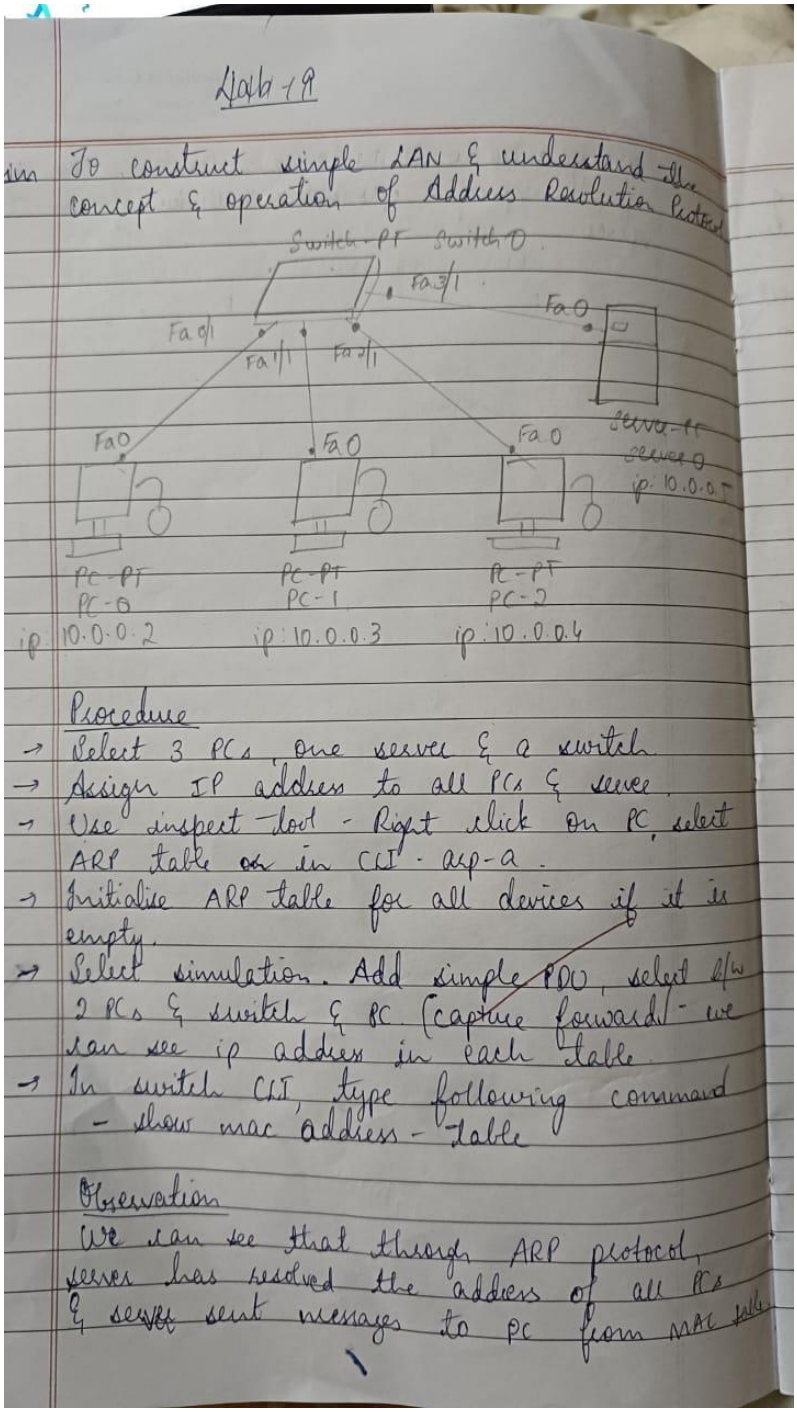


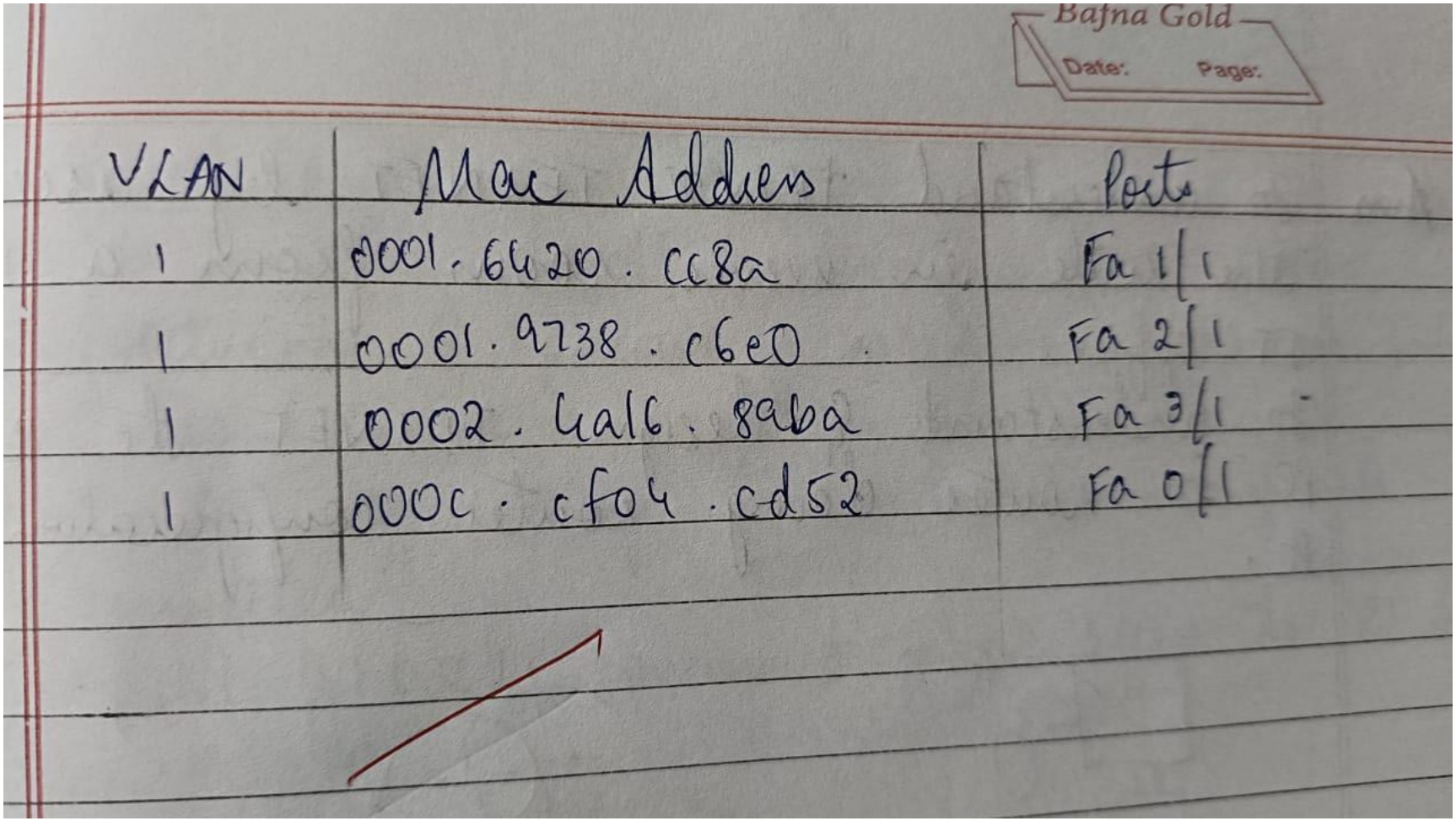
TOPOLOGY:


Switch-PT
Switch0
PC-PT
PC0
Server-PT
Server0

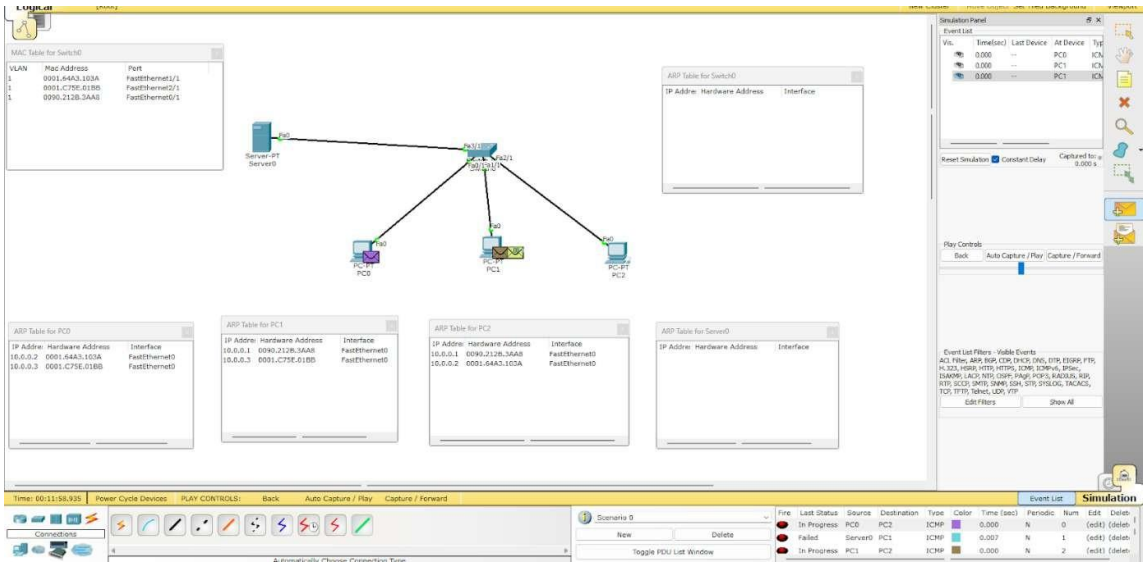

Web Browser
< > URL http://cv
Ganshree resume
Quick Links:
linkdin
github
portfolio
username
education
bms college of engineering 8.8.6 cgpa computrer science engineering
project
loibrary management system

PROGRAM9: To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)
OBSERVATION:



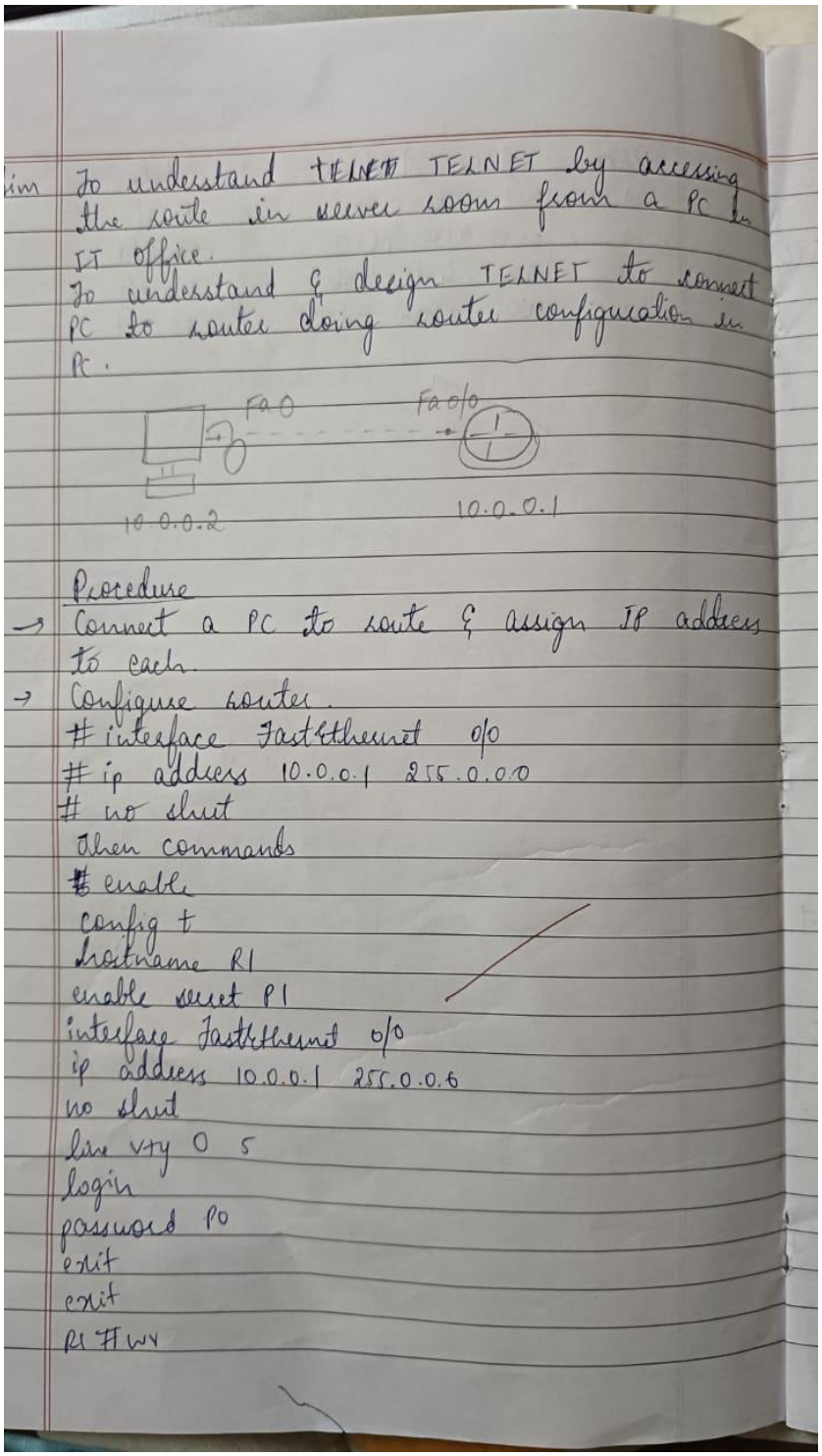

Bafna Gold
Date:
Page:
VLAN
Mac Addiess
Ports
1
0001.6420.cc8a
Fa 1/1
1
0001.9738.c6e0
Fa 2/1
1
0002.4a16.89ba
Fa 3/1
1
000c.cf04.cd52
Fa 0/1

TOPOLOGY:



PROGRAM10: To understand the operation of TELNET by accessing the router in server room from a PC in IT office.

OBSERVATION:



Bafna Gold
Date: Page:

Observation

It is observed that through TELNET the hostname & password is given to access CLI of router in any other device.

In PC type ping first to know if its connected.

telnet 10.0.0.1 command gives access to R1 host in PC P0.

User access verification

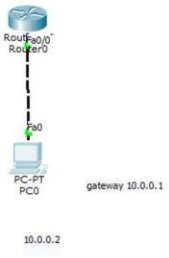
Password: P0

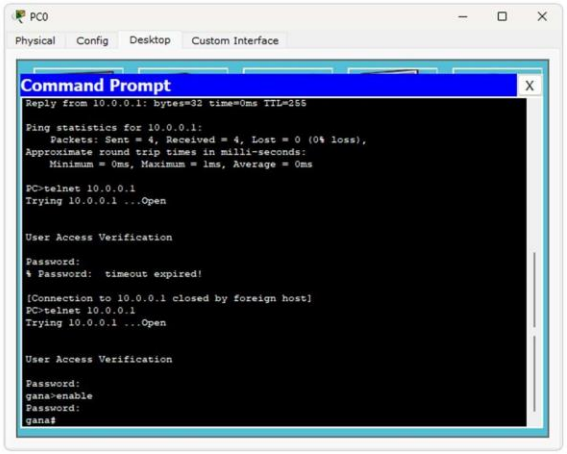
R1 > enable

Password: P1

R1 #

TOPOLOGY and OUPUT:

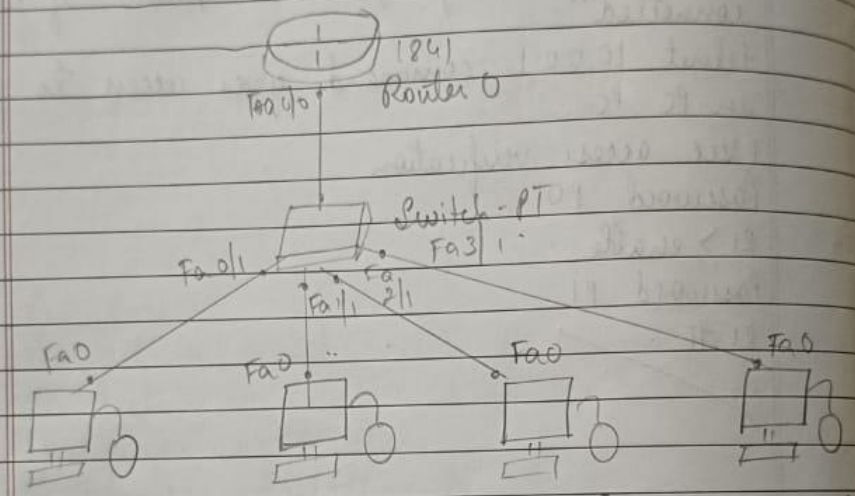




PROGRAM11: To construct a VLAN and make the PC's communicate among a VLAN
OBSERVATION:

im Creating a new VLAN & make the PCs communicate among VLAN

To configure VLAN & make PC's communicate with each other.



ip	192.168.1.2	192.168.1.3	192.168.2.2	192.168.2.3
gway	192.168.1.1	192.168.1.1	192.168.2.1	192.168.2.1

Procedure

→ Take a router, connect it to a switch & connect all PC's to switch PT switch

→ Configure the 2 PC's ip address 192.168.1.2 192.168.1.3 & gateway 192.168.1.1 & other 2 PC's ip addres 192.168.2.2 192.168.2.3 & gateway 192.168.2.2

→ Go to router configure first 2 PCs in the router commands.

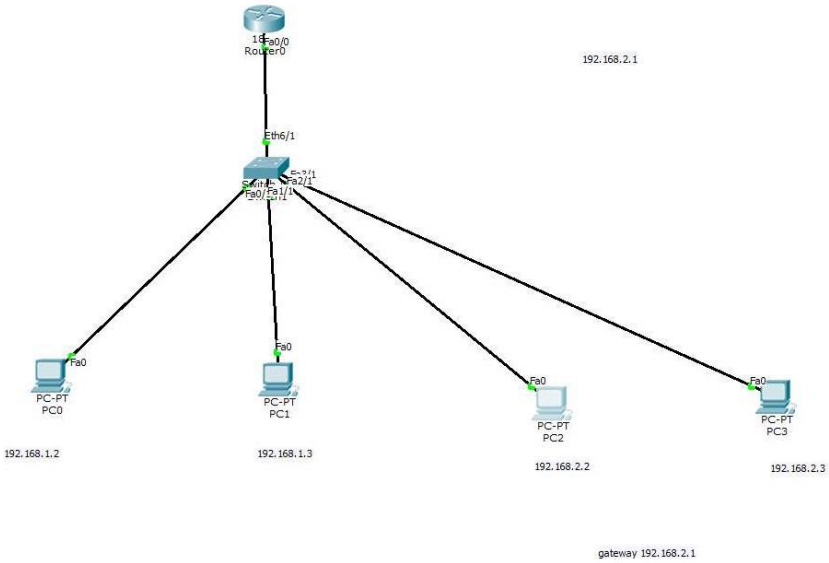
enable
config t
interface FastEthernet 0/0
ip address 192.168.1.1 255.255.255.0
no shut
exit

Bafna Gold
Date: Page:

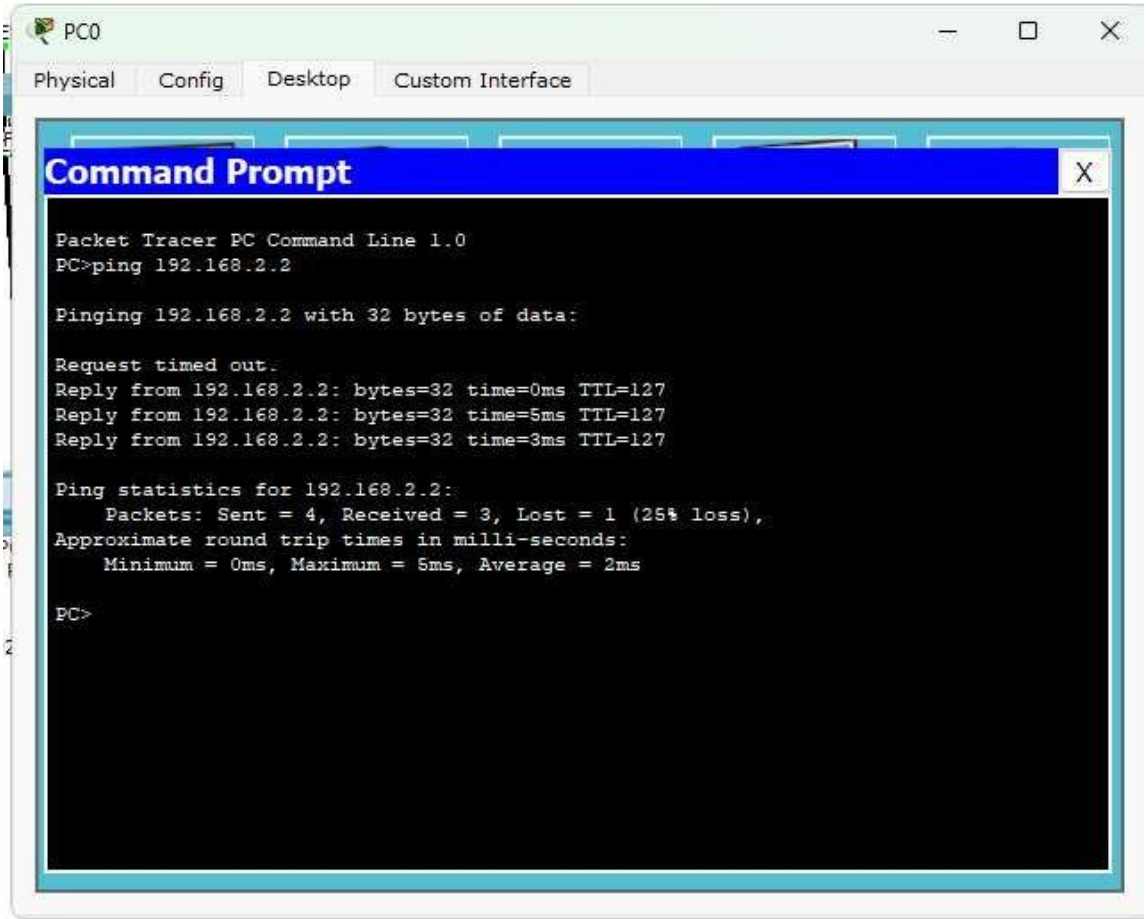
→ Then select switch, go to configure, tap select VLAN Database.
→ Set VLAN number to 2 & name press add.
→ Do VLAN trunking
→ Go to FastEthernet 0/1, select trunk.
→ Go to FastEthernet 2/1, selet VLAN name.
→ In router CLI, next VLAN Database & enter the number & VLAN created.
Go to CLI - commands
Router(VLAN) # exit
Router # config t
interface FastEthernet 0/0
Router(config-subif) # encapsulation dot1q 2
ip address 192.168.2.1 255.255.255.0
no shut
exit
exit
→ Ping devices from first two routers to the other two.

Observation
VLAN tracking allows switches to forward from different VLAN, over single link called trunk. This is done by adding an additional header information called tag to the ethernet frame. The process of adding this small header is called VLAN tagging.
The router switch understand VLAN.
This virtual LAN client & 2 LAN can be using VLAN for different IP route.

TOPOLOGY:

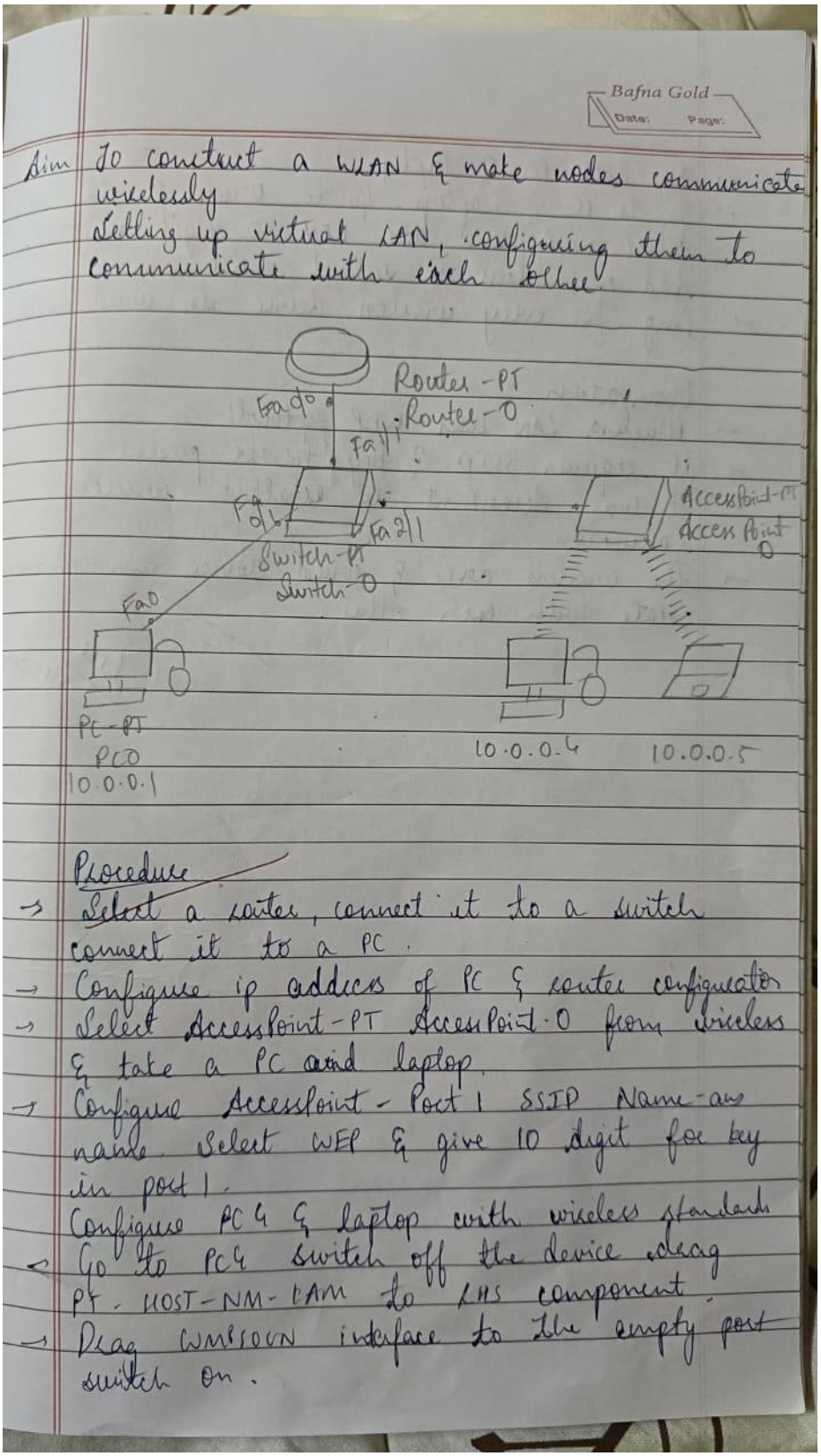


OUTPUT:



PROGRAM12: To construct a WLAN and make the nodes communicate wirelessly To construct a WLAN and make the nodes communicate wirelessly

OBSERVATION:

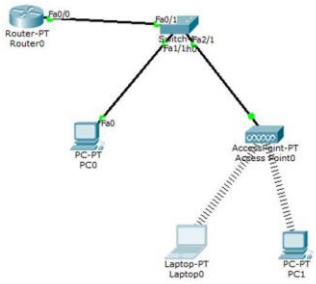


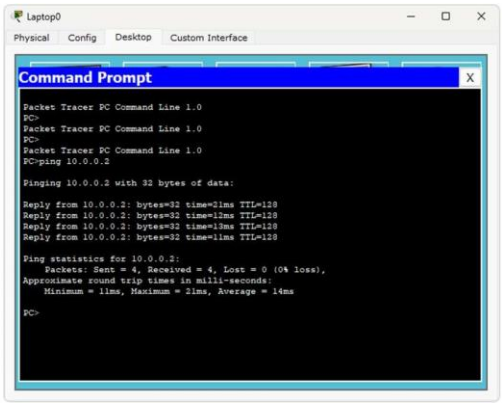
→ In both PC & laptop.
→ Go to PC & Laptop. Go to Wireless O Add one.
Select WEP and add KEY 10 digit number.
Add ip address & subnet mask.
→ Ping for every wireless device to wired device

Observation
→ Wireless LAN uses WEP protocol.
→ It requires SSID & key to be present.
→ It uses Accesspoint to establish wireless connection.
→ The wireless device & wired device can communicate with each other.

Lee
5/12/24

TOPOLOGY AND OUPUT

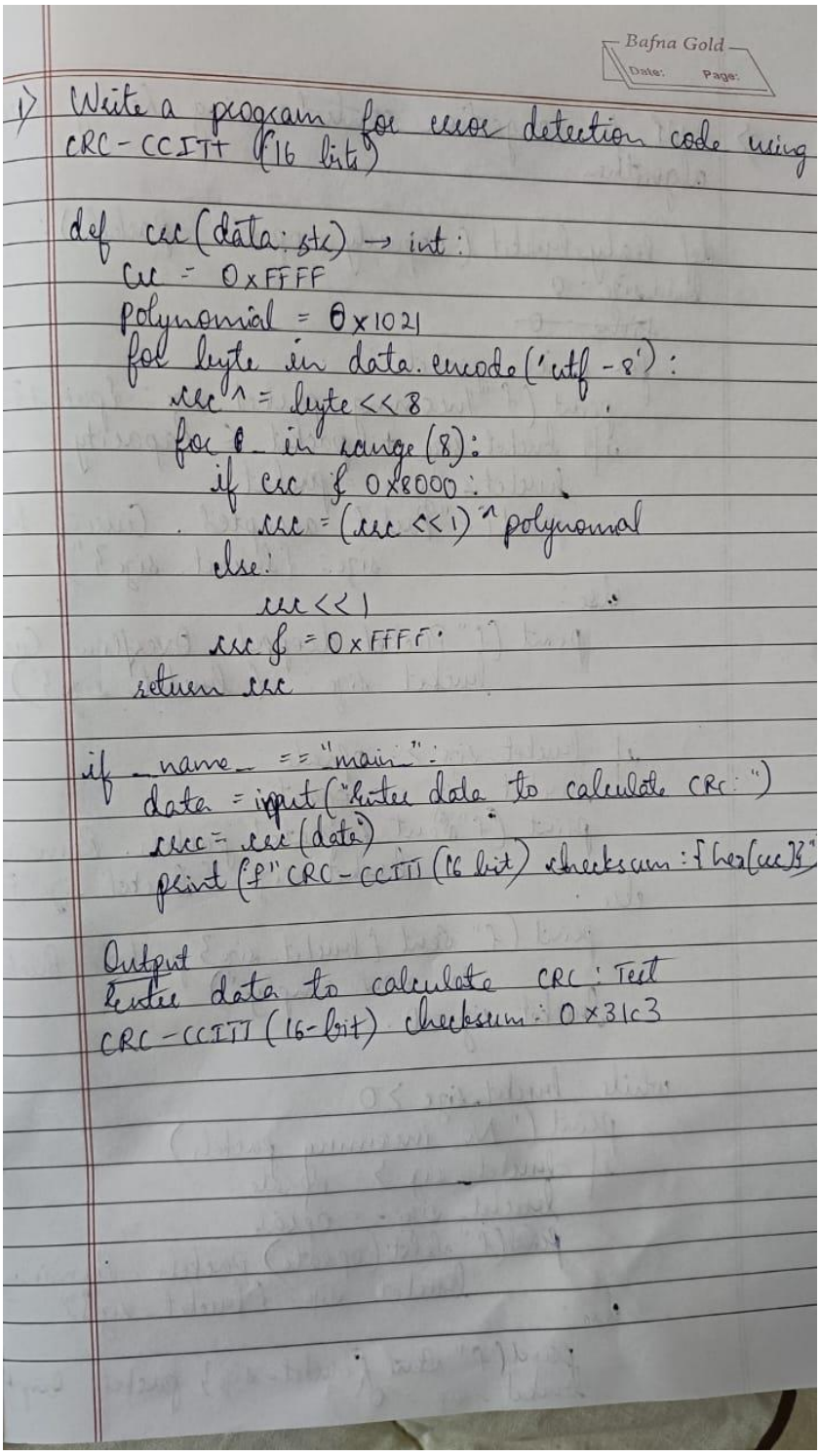




CYCLE-2

PROGRAM1: Write a program for error detecting code using CRC-CCITT (16-bits)

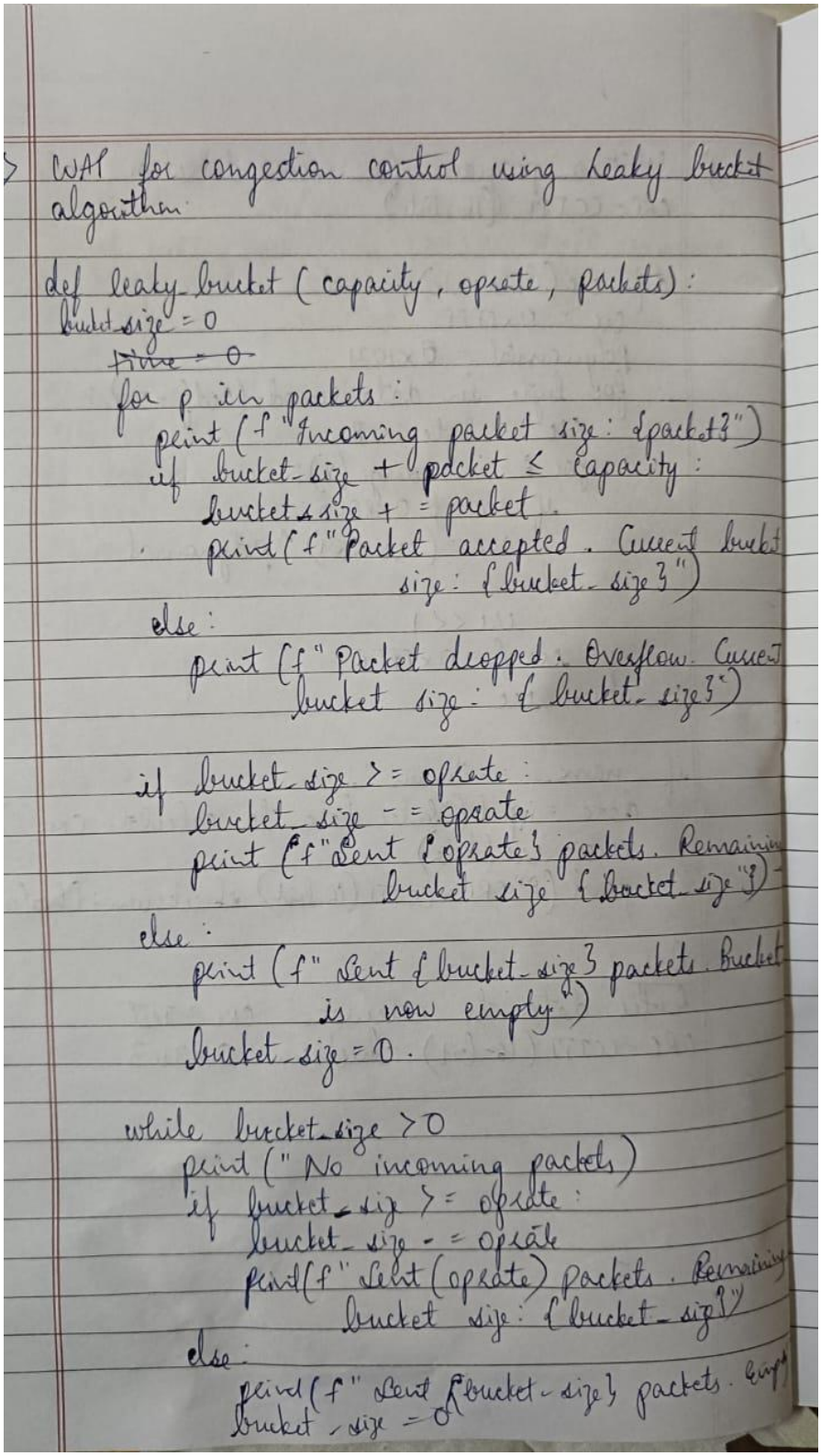
OBSERVATION:



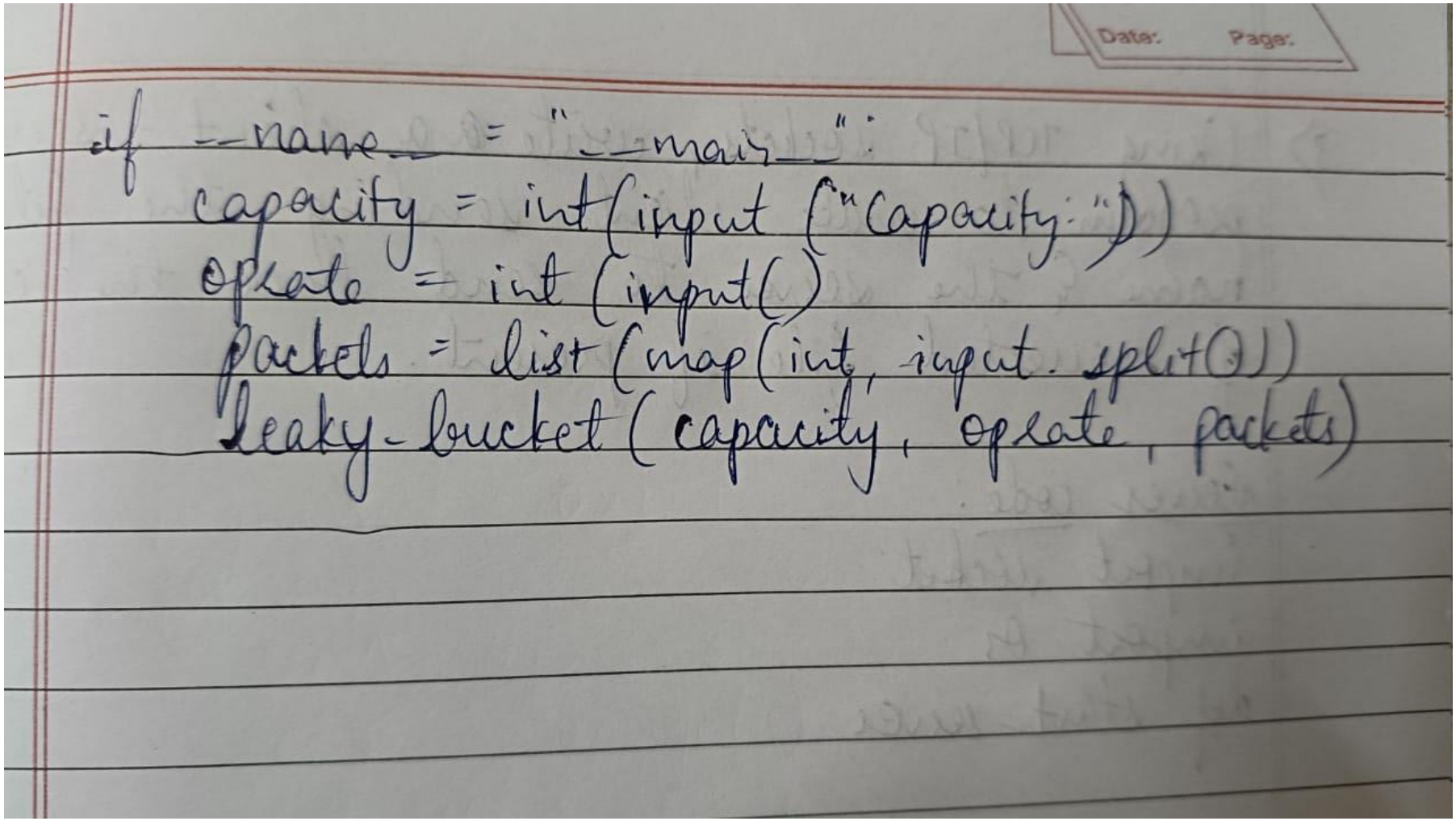
```
def crc_ccitt(data: str, polynomial: int = 0x1021, init_value: int = 0xFFFF) -> int:
    """
    Calculate CRC-CCITT (16-bit) checksum for the given data.

    Args:
        data (str): Input data to calculate CRC for.
        polynomial (int): CRC polynomial, default is 0x1021.
        init_value (int): Initial value of CRC register, default is 0xFFFF.

    Returns:
        int: Calculated CRC value.
    """
    crc = init_value
    for byte in data.encode('utf-8'):
        crc ^= byte << 8
        for _ in range(8):
            if crc & 0x8000:
                crc = (crc << 1) ^ polynomial
            else:
                crc <<= 1
            crc &= 0xFFFF  # Ensure CRC remains 16-bit
    return crc


# Example usage
if __name__ == "__main__":
    input_data = input("Enter the data to calculate CRC: ")
    crc_value = crc_ccitt(input_data)
    print(f"CRC-CCITT (16-bit) value: {hex(crc_value)}")
```

```
Enter the data to calculate CRC: Test
CRC-CCITT (16-bit) value: 0x2888
```

PROGRAM2: Write a program for congestion control using Leaky bucket algorithm.
OBSERVATION:



```
WAP for congestion control using Leaky bucket
algorithm

def leaky_bucket (capacity, operate, packets):
    bucket_size = 0
    ~~time = 0~~
    for p in packets:
        print (f"Incoming packet size: {packet}")
        if bucket_size + packet <= capacity:
            bucket_size += packet
            print (f"Packet accepted. Current bucket
                    size: {bucket_size}")
        else:
            print (f"Packet dropped. Overflow. Current
                    bucket size: {bucket_size}")

    if bucket_size >= operate:
        bucket_size -= operate
        print (f"Sent {operate} packets. Remaining
                bucket size {bucket_size}")
    else:
        print (f"Sent {bucket_size} packets. Bucket
                is now empty")
        bucket_size = 0.

    while bucket_size > 0
        print ("No incoming packets)
        if bucket_size >= operate:
            bucket_size -= operate
            print(f"Sent (operate) packets. Remaining
                    bucket size: {bucket_size}")
        else:
            print(f"Sent {bucket_size} packets. Empty
            bucket_size = 0
```



```
if __name__ = "__main__":
    capacity = int(input("Capacity: "))
    operate = int(input())
    packets = list(map(int, input.split()))
    leaky_bucket(capacity, operate, packets)
```

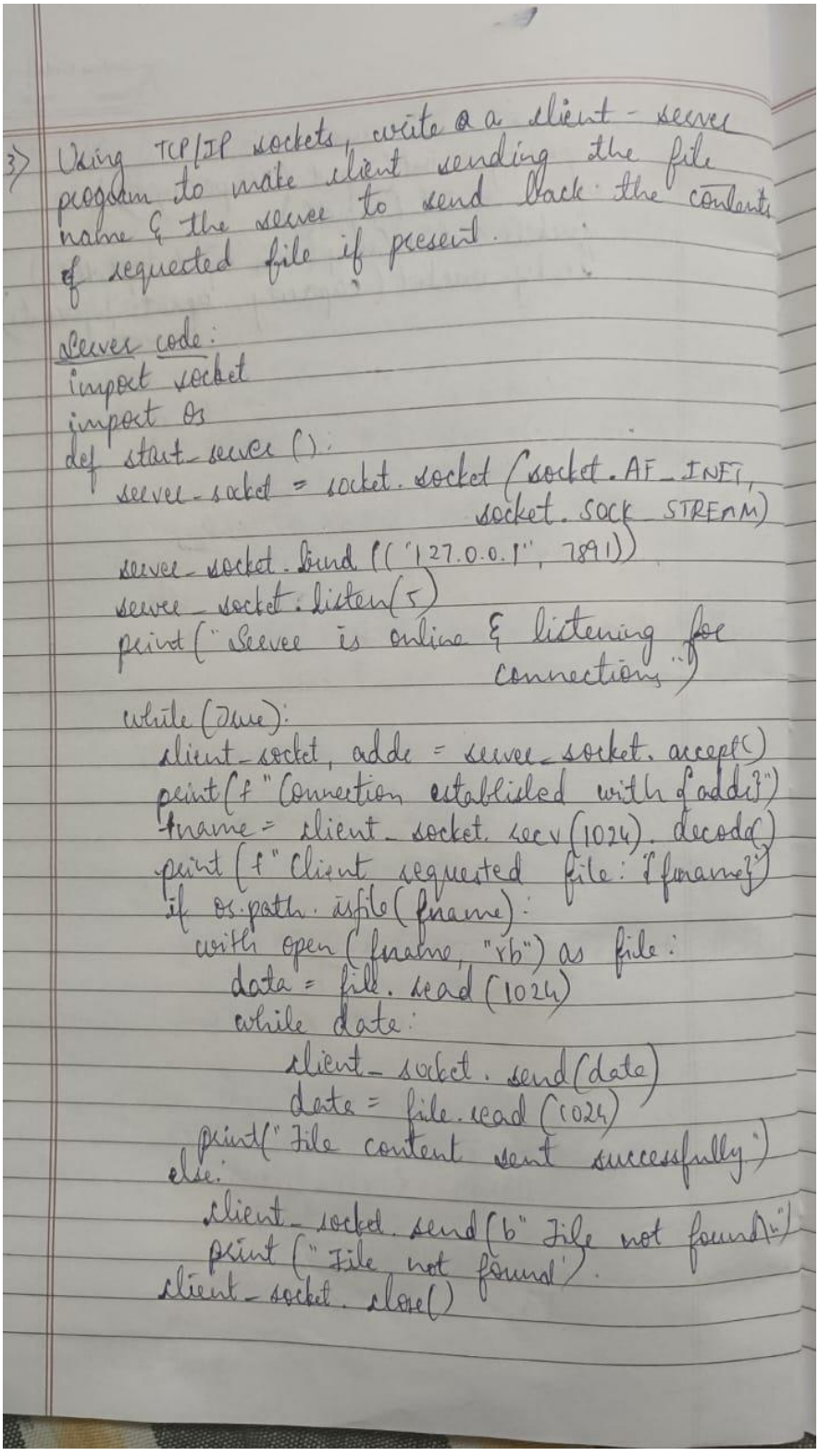
CODE:

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h> // for sleep function
#define NOF_PACKETS 5
// Function to simulate sending packets
void send_packet(int packet_size, int output_rate) {
    while (packet_size > 0) {
        int sent = (packet_size < output_rate) ? packet_size : output_rate;
        printf("Packet of size %d Transmitted---", sent);
        packet_size -= sent;
        printf("Bytes Remaining to Transmit: %d\n", packet_size);
        sleep(1);  // Simulate time delay between packets
    }
}

int main() {
    int output_rate, bucket_size, incoming_packet_size;
    int i, packet_size[NOF_PACKETS];

    // Input number of packets and their sizes
    for(i = 0; i < NOF_PACKETS; i++) {
        packet_size[i] = rand() % 100;  // Random packet size between 0 and
printf("packet[%d]:%d bytes\n", i, packet_size[i]);
    }

    printf("Enter the Output rate:");
    scanf("%d", &output_rate);

    printf("Enter the Bucket Size:");
    scanf("%d", &bucket_size);

    for(i = 0; i < NOF_PACKETS; i++) {
        printf("\nIncoming Packet size: %d\n", packet_size[i]);
        if(packet_size[i] > bucket_size) {
            printf("Incoming packet size (%dbytes) is Greater than bucket capacity (%dbytes)-
PACKET REJECTED\n", packet_size[i], bucket_size);
            continue;
        }

        printf("Bytes remaining to Transmit: %d\n", packet_size[i]);
        send_packet(packet_size[i], output_rate);
    }
    return 0;
}
```

```
packet[0]:83 bytes
packet[1]:86 bytes
packet[2]:77 bytes
packet[3]:15 bytes
packet[4]:93 bytes
Enter the Output rate:50
Enter the Bucket Size:300

Incoming Packet size: 83
Bytes remaining to Transmit: 83
Packet of size 50 Transmitted---Bytes Remaining to Transmit: 33
Packet of size 33 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 86
Bytes remaining to Transmit: -86
Packet of size 50 Transmitted---Bytes Remaining to Transmit: 36
Packet of size 36 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 77
Bytes remaining to Transmit: 77
Packet of size 50 Transmitted---Bytes Remaining to Transmit: 27
Packet of size 27 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 15
Bytes remaining to Transmit: 15
Packet of size 15 Transmitted---Bytes Remaining to Transmit: 0

Incoming Packet size: 93
Bytes remaining to Transmit: 93
Packet of size 50 Transmitted---Bytes Remaining to Transmit: 43
Packet of size 43 Transmitted---Bytes Remaining to Transmit: 0


=== Code Execution Successful ===
```

PROGRAM3: 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.

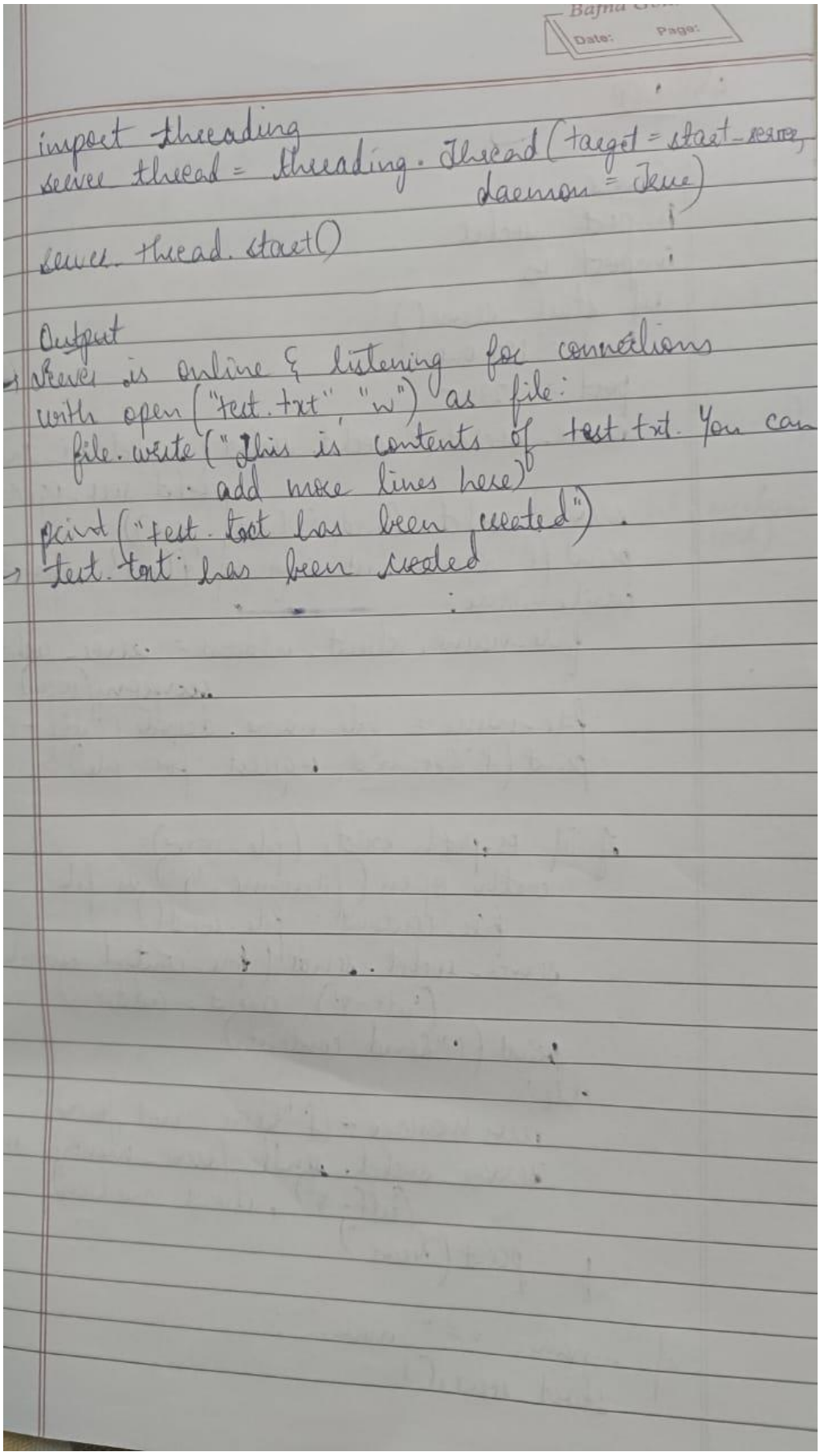
OBSERVATION:



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

Server code:

```
import socket
import os
def start_server():
    server_socket = socket.socket(socket.AF_INET,
                                  socket.SOCK_STREAM)
    server_socket.bind(('127.0.0.1', 7891))
    server_socket.listen(5)
    print("Server is online & listening for
                                  connections...")
    while(True):
        client_socket, addr = server_socket.accept()
        print(f"Connection established with {addr}")
        fname = client_socket.recv(1024).decode()
        print(f"Client requested file: {fname}")
        if os.path.isfile(fname):
            with open(fname, "rb") as file:
                data = file.read(1024)
                while data:
                    client_socket.send(data)
                    data = file.read(1024)
            print("File content sent successfully")
        else:
            client_socket.send(b"File not found!")
            print("File not found")
        client_socket.close()
```



```
import threading
server_thread = threading.Thread(target=start_server,
                                 daemon=True)
server_thread.start()
```

Output

```
Server is online & listening for connections
with open("test.txt", "w") as file:
    file.write("This is contents of test.txt. You can
        add more lines here")
print("test.txt has been created")
```

test.txt has been created

CODE:

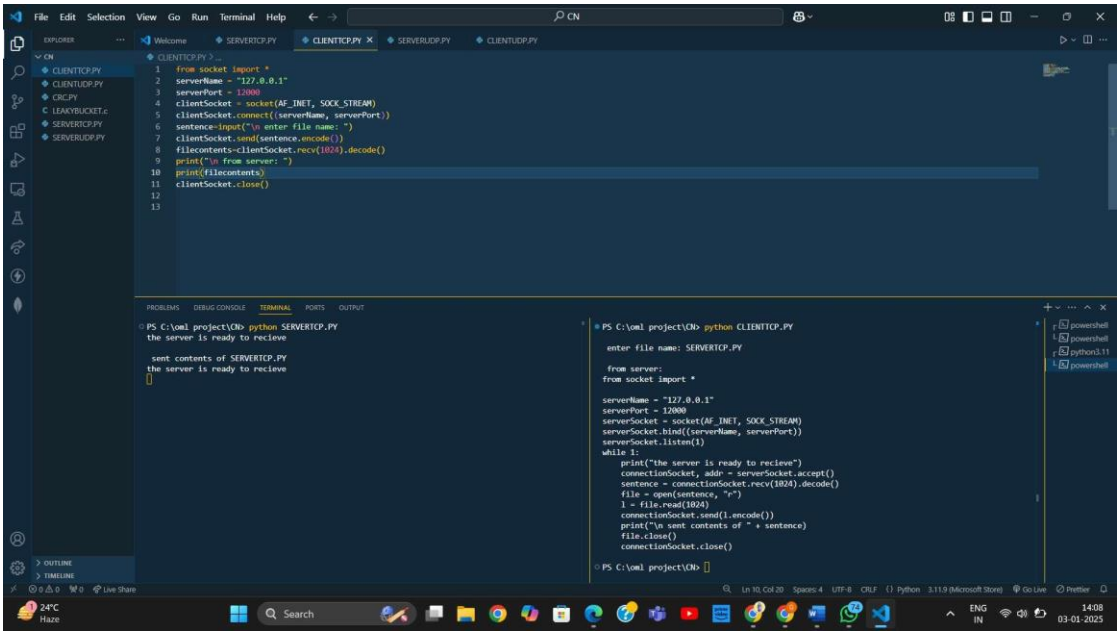
SERVERTCP.PY:

```
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("the server is ready to recieve")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file = open(sentence, "r")
    l = file.read(1024)
    connectionSocket.send(l.encode())
    print("\n sent contents of " + sentence)
    file.close()
    connectionSocket.close()
```

CLIENTTCP.PY:

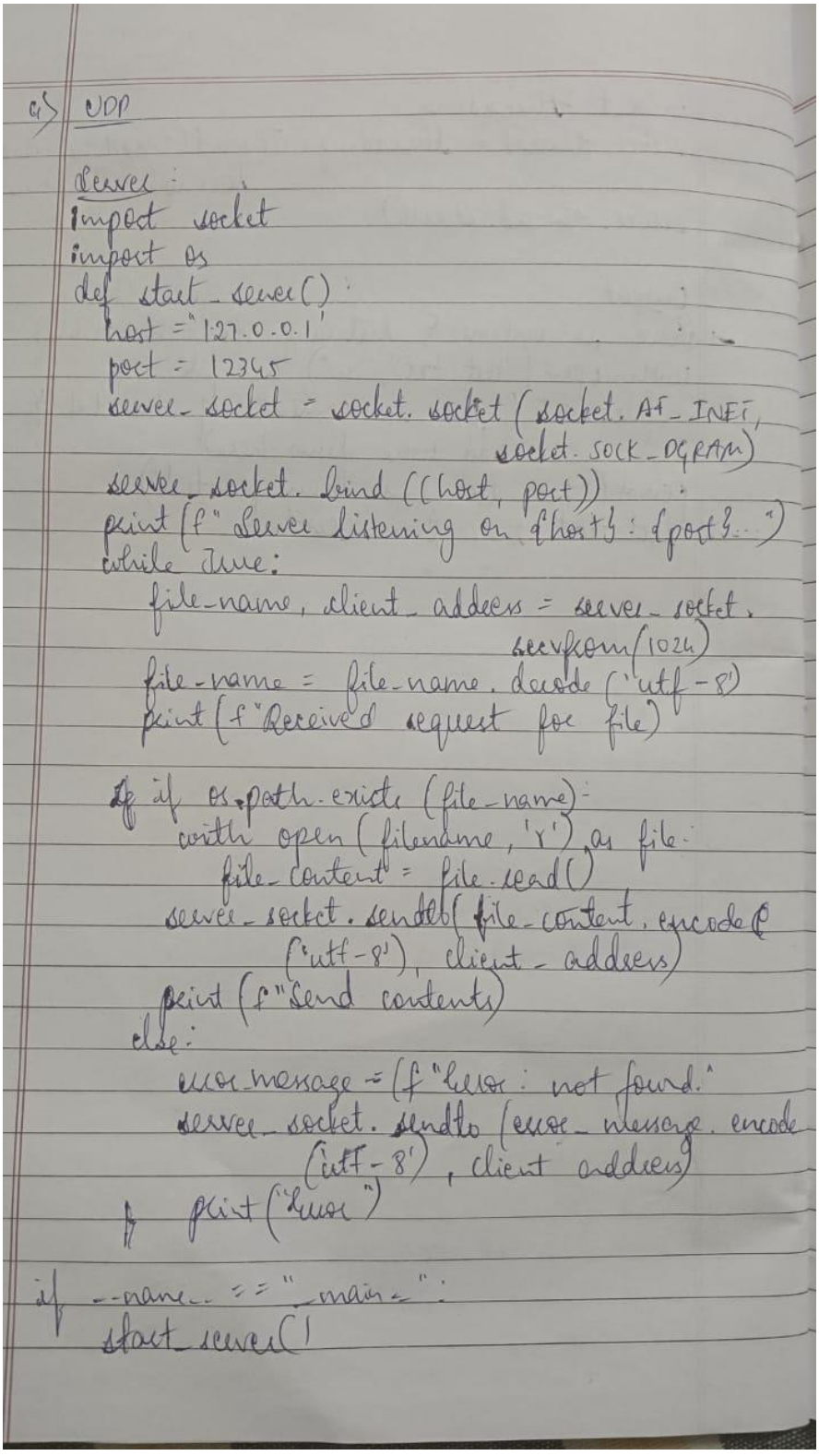
```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence=input("\n enter file name: ")
clientSocket.send(sentence.encode())
filecontents=clientSocket.recv(1024).decode()
print("\n from server: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

OUTPUT:


CLIENTTCP.PY
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence=input("\n enter file name: ")
clientSocket.send(sentence.encode())
filecontents=clientSocket.recv(1024).decode()
print("\n from server: ")
print(filecontents)
clientSocket.close()
PS C:\oml project\CN> python SERVERTCP.PY
the server is ready to recieve
sent contents of SERVERTCP.PY
the server is ready to recieve
PS C:\oml project\CN> python 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 recieve")
connectionSocket, addr = serverSocket.accept()
sentence = connectionSocket.recv(1024).decode()
file = open(sentence, "r")
l = file.read(1024)
connectionSocket.send(l.encode())
print("\n sent contents of " + sentence)
file.close()
connectionSocket.close()
PS C:\oml project\CN>

PROGRAM4: 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:



```
6) UDP

Server:
import socket
import os
def start_server():
    host = '127.0.0.1'
    port = 12345
    server_socket = socket.socket(socket.AF_INET,
                                  socket.SOCK_DGRAM)
    server_socket.bind((host, port))
    print(f"Server listening on {host}:{port}...")
    while True:
        file_name, client_address = server_socket.
                                    recvfrom(1024)
        file_name = file_name.decode('utf-8')
        print(f"Received request for file)

        if os.path.exists(file_name):
            with open(filename, 'r') as file:
                file_content = file.read()
            server_socket.sendto(file_content.encode(
                ('utf-8'), client_address)
            print(f"Send contents)
        else:
            error_message = (f"Error: not found."
            server_socket.sendto(error_message.encode
                (utf-8'), client address)
        print("Error")

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

Client

```
import socket
def start_client():
    host = '127.0.0.1'
    port = 12345
    client_socket = socket.socket(socket.AF_INET,
                                  socket.SOCK_DGRAM)
    file_name = input()
    client_socket.sendto(filename.encode('utf-8'
                         (host, port))
    response, server_address = client_socket.recvfro
                               (1024)
    print("Response")
    print(response.decode('utf-8'))
    client_socket.close()

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

CODE:

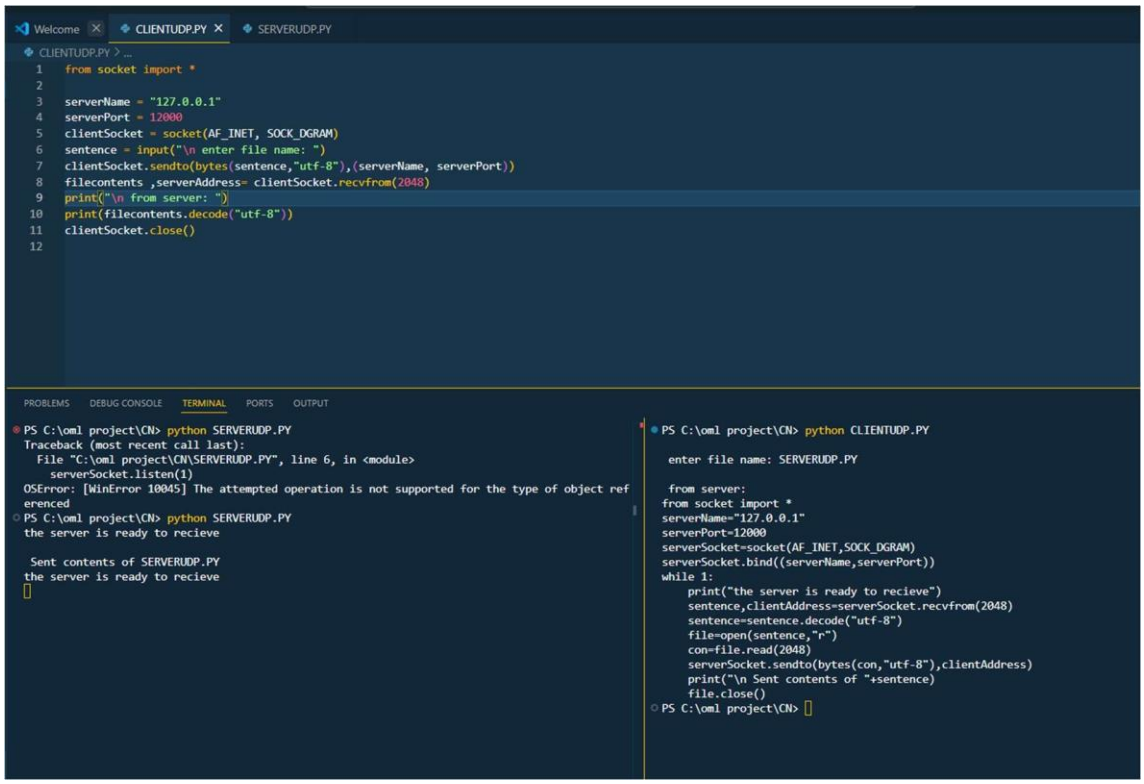
SERVERUDP.PY

```
from socket import *
serverName="127.0.0.1"


serverPort=12000
serverSocket=socket(AF_INET,SOCK_DGRAM)
serverSocket.bind((serverName,serverPort))
while 1:
    print("the server is ready to recieve")
    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("\n Sent contents of "+sentence)
    file.close()
```

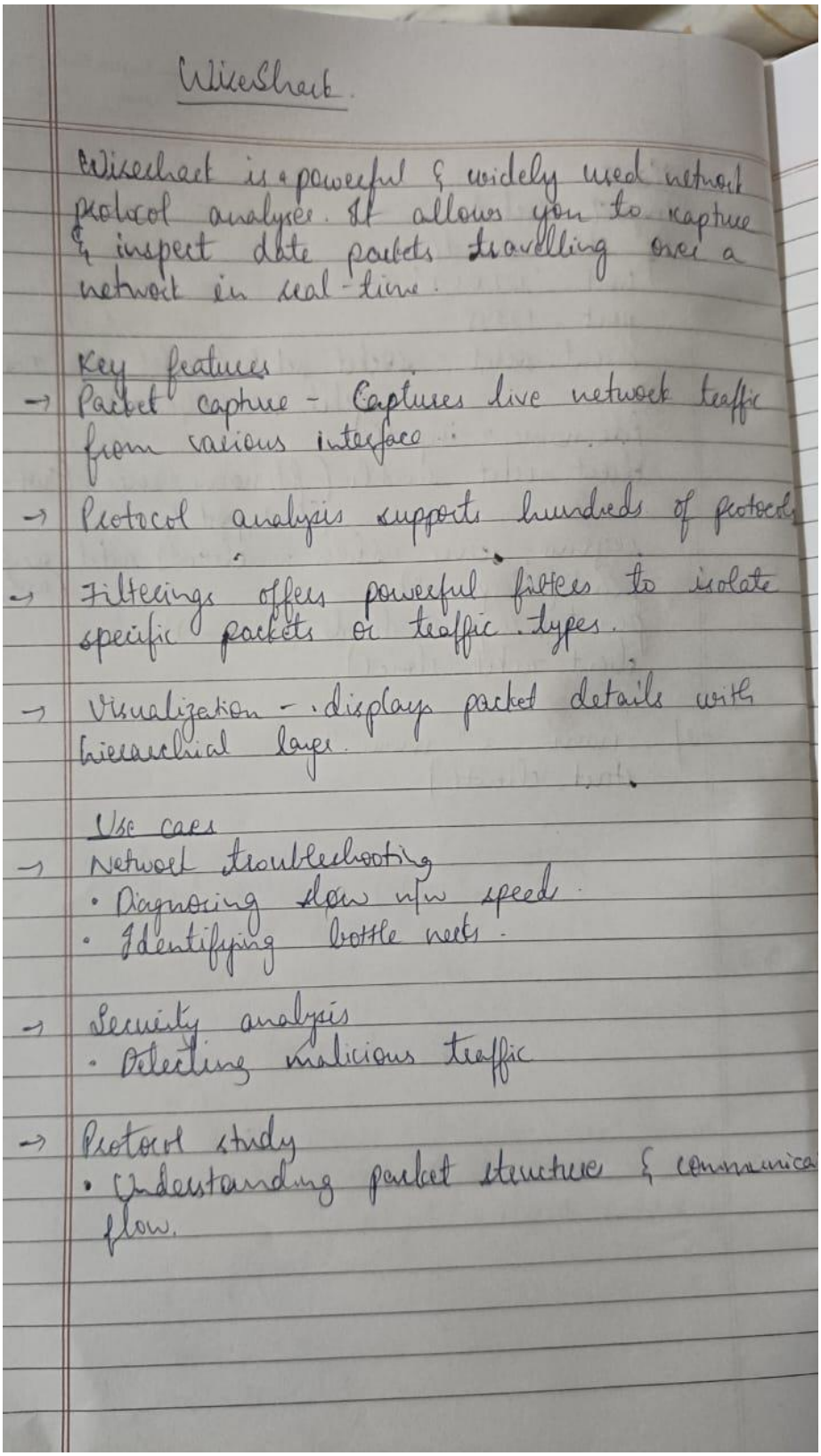
CLIENTUDP.PY:

```
from socket import *

serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("\n enter file name: ")
clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))
filecontents ,serverAddress= clientSocket.recvfrom(2048)
print("\n from server: ")
print(filecontents.decode("utf-8"))
clientSocket.close()
```

OUTPUT:


Welcome
CLIENTUDP.PY
SERVERUDP.PY
CLIENTUDP.PY > ...
1 from socket import *
2
3 serverName = "127.0.0.1"
4 serverPort = 12000
5 clientSocket = socket(AF_INET, SOCK_DGRAM)
6 sentence = input("\n enter file name: ")
7 clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))
8 filecontents ,serverAddress= clientSocket.recvfrom(2048)
9 print("\n from server: ")
10 print(filecontents.decode("utf-8"))
11 clientSocket.close()
12
PROBLEMS
DEBUG CONSOLE
TERMINAL
PORTS
OUTPUT
PS C:\oml project\CN> python SERVERUDP.PY
Traceback (most recent call last):
File "C:\oml project\CN\SERVERUDP.PY", line 6, in <module>
serverSocket.listen(1)
OSError: [WinError 10045] The attempted operation is not supported for the type of object referenced
PS C:\oml project\CN> python SERVERUDP.PY
the server is ready to recieve
Sent contents of SERVERUDP.PY
the server is ready to recieve
PS C:\oml project\CN> python CLIENTUDP.PY
enter file name: SERVERUDP.PY
from server:
from socket import *
serverName="127.0.0.1"
serverPort=12000
serverSocket=socket(AF_INET,SOCK_DGRAM)
serverSocket.bind((serverName,serverPort))
while 1:
print("the server is ready to recieve")
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("\n Sent contents of "+sentence)
file.close()
PS C:\oml project\CN>

WIRESHARK:



Wireshark.

Wireshark is a powerful & widely used network protocol analyser. It allows you to capture & inspect data packets travelling over a network in real-time.

Key features
→ Packet capture - Captures live network traffic from various interface.
→ Protocol analysis supports hundreds of protocols.
→ Filtering offers powerful filters to isolate specific packets or traffic types.
→ Visualization - displays packet details with hierarchial layer.

Use cases
→ Network troubleshooting
• Diagnosing slow n/w speeds.
• Identifying bottle necks.
→ Security analysis
• Detecting malicious traffic
→ Protocol study
• Understanding packet structure & communica flow.