BMS COLLEGE OF ENGINEERING BENGALURU

Autonomous Institute, Affiliated to VTU

A Lab Report on

Computer Networks Lab

Submitted in partial fulfillment for the award of the degree of

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by:

Prakhyati Bansal
1BM21CS136

Under the Guidance of:

Prof. Swathi Sridharan
Assistant Professor
BMSCE

Department of Computer Science and Engineering
BMS College of Engineering
Bull Temple Road, Basavanagudi, Bangalore 560019

2022-2023

CERTIFICATE

This is to certify that the Lab work entitled "COMPUTER NETWORKS LAB" carried out by **Prakhyati Bansal (1BM21CS136),** 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 2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **COMPUTER NETWORKS - (22CS4PCCON) work prescribed for the said degree.**

Prof. Swathi Sridharan
Assistant Professor
Department of CSE
BMSCE, Bengaluru

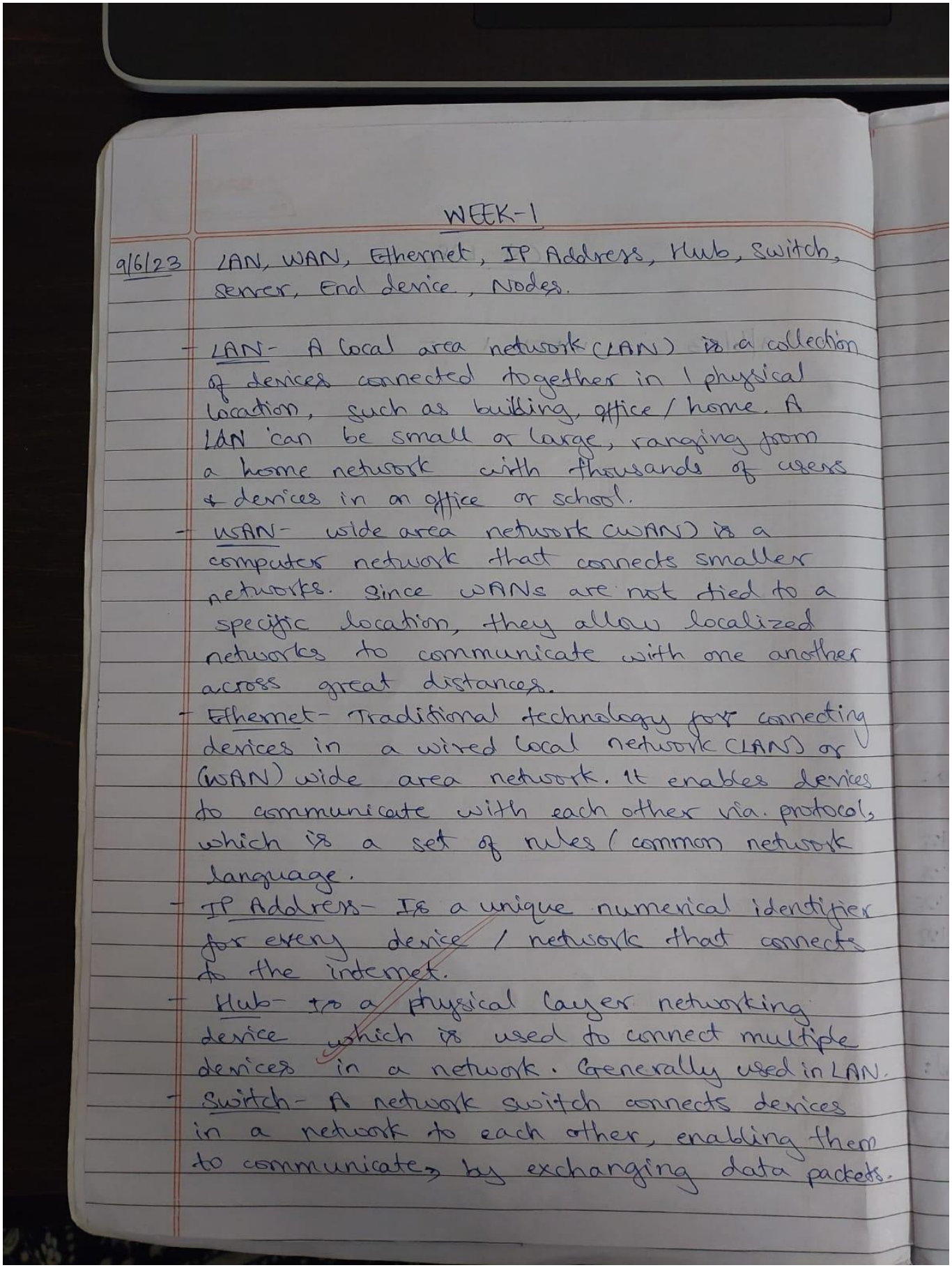
Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

INDEX

LAB 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-



WEEK-1

9/6/23 LAN, WAN, Ethernet, IP Address, Hub, Switch, Server, End device, Nodes.

- LAN- A local area network (LAN) is a collection of devices connected together in 1 physical location, such as building, office / home. A LAN can be small or large, ranging from a home network with thousands of users & devices in an office or school.
- WAN- wide area network (WAN) is a computer network that connects smaller networks. Since WANs are not tied to a specific location, they allow localized networks to communicate with one another across great distances.
- Ethernet- Traditional technology for connecting devices in a wired local network (LAN) or (WAN) wide area network. It enables devices to communicate with each other via. protocol, which is a set of rules / common network language.
- IP Address- Is a unique numerical identifier for every device / network that connects to the internet.
- Hub- is a physical layer networking device which is used to connect multiple devices in a network. Generally used in LAN.
- Switch- A network switch connects devices in a network to each other, enabling them to communicate by exchanging data packets.

witch,

ollection
sal
A
om
users

r
o a
ed
nother

necting
) or
devices
otocol,
ork

tiplier
ects

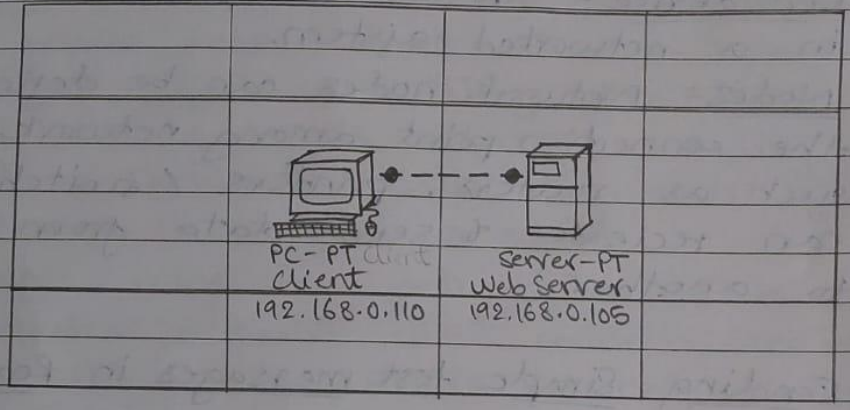
ple
LAN.
es
hem
ckets.

- Server- A piece of computer hardware or software that provides functionality for other programs / devices called 'clients'.
- End device - A source / destination device in a networked system.
- Nodes- Network nodes can be defined as the connection point among network devices such as routers, printers / switches that can recieve & send data from 1 endpoint to another.

- Sending Simple test messages in Realtime-

1) Add a client end-device & a web server end-device.
2) Connect both using a copper cross-over cable.
3) set the client's DNS server to 192.168.0.105. Set the IP address under fast ethernet to 192.168.0.110.
4) select webserver & IP address to be set to 192.168.0.105.
5) select the DNS services & set the domain name as "www.firstlab.com" & IP address as 192.168.0.105 & add.
6) Ensure DNS server is on.
7) Add simple PDU tool is used to send a simple 1-time message from PC to server & vice-versa.
8) The log values are displayed on the PDU list window.

- Topology -



- command Prompt -

PC > ping 192.168.0.110
pinging 192.168.0.110 with 32 bytes of data:

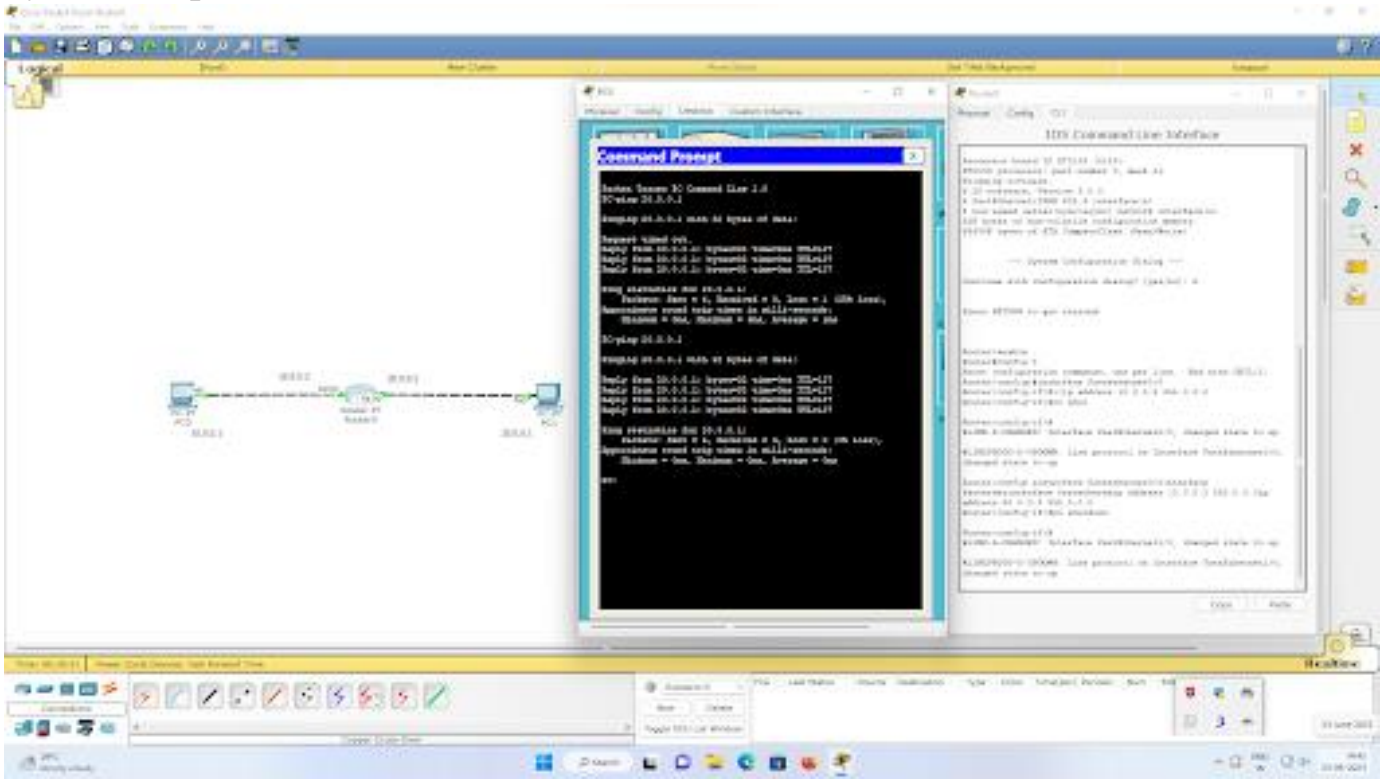
Reply from 192.168.0.110: bytes=32 time=4ms TTL=128
Reply from 192.168.0.110: bytes=32 time=2ms TTL=128
Reply from 192.168.0.110: bytes=32 time=4ms TTL=128
Reply from 192.168.0.110: bytes=32 time=4ms TTL=128

Ping statistics for 192.168.0.110:
Packets: sent=4, recieved=4, Loss=0 (0% Loss),
approximate round trip times in millisecond:

minimum=2ms, maximum=4ms, average=3ms

Thu 23/6/22

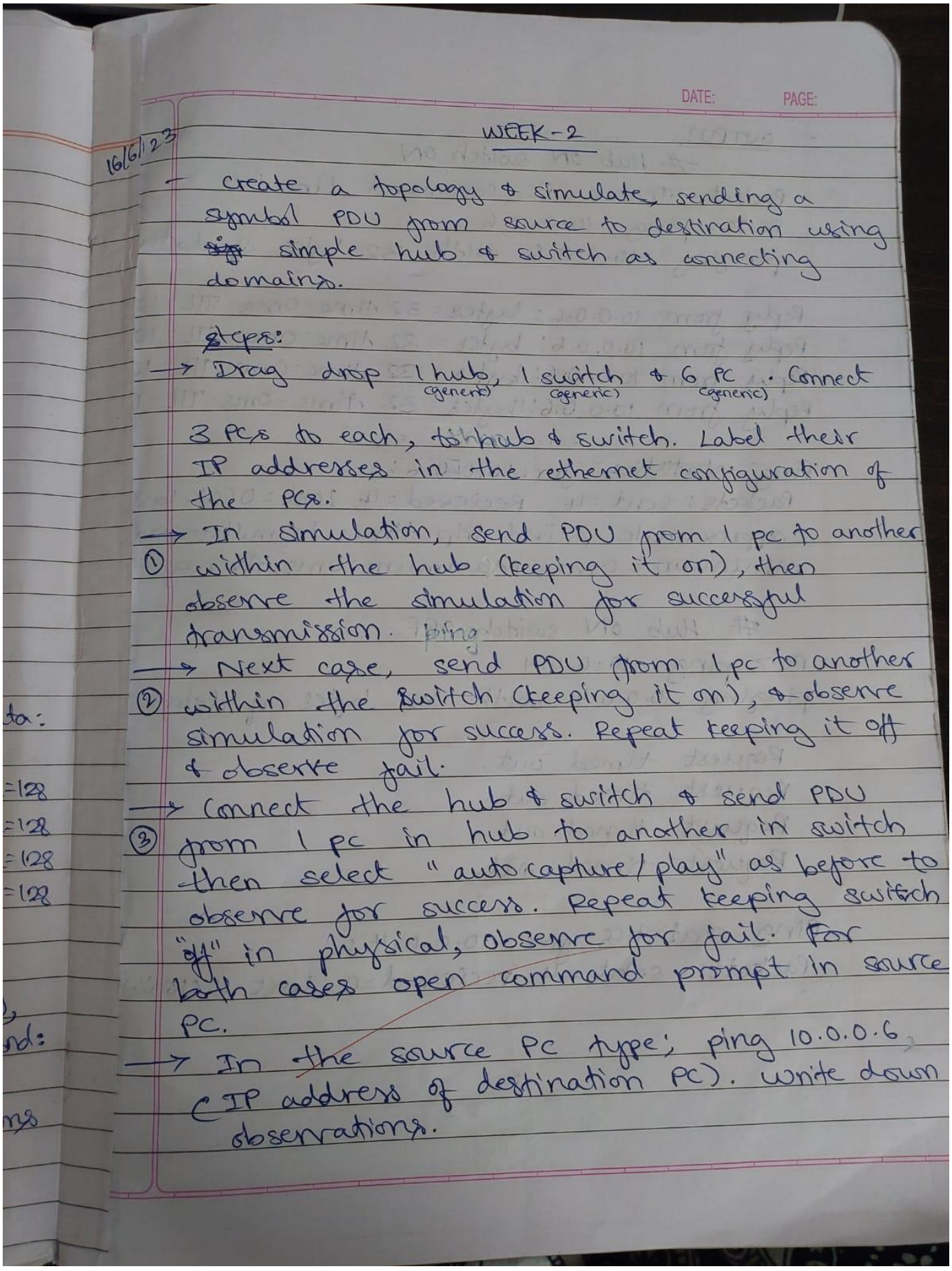
System Output-



LAB 2

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

Observation-



DATE: PAGE:

16/6/23

WEEK-2

- Create a topology & simulate, sending a simple PDU from source to destination using simple hub & switch as connecting domains.

Steps:

→ Drag drop 1 hub (generic), 1 switch (generic) & 6 PC (generic). Connect 3 PCs to each, to hub & switch. Label their IP addresses in the ethernet configuration of the PCs.

→ In simulation, send PDU from 1 pc to another ① within the hub (keeping it on), then observe the simulation for successful transmission. ping

→ Next case, send PDU from 1 pc to another ② within the switch (keeping it on), & observe simulation for success. Repeat keeping it off & observe fail.

→ Connect the hub & switch & send PDU ③ from 1 pc in hub to another in switch then select "auto capture / play" as before to observe for success. Repeat keeping switch "off" in physical, observe for fail. For both cases open command prompt in source PC.

→ In the source PC type; ping 10.0.0.6 (IP address of destination PC). write down observations.

DATE: PAGE:

- OUTPUT:-

Hub ON switch ON

Packet Tracer PC Command Line 1.0
PC> ping 10.0.0.6
pinging 10.0.0.6 with 32 bytes of data:

Reply from 10.0.0.6: bytes=32 time=0ms TTL=128
Reply from 10.0.0.6: bytes=32 time=0ms TTL=128
Reply from 10.0.0.6: bytes=32 time=0ms TTL=128
Reply from 10.0.0.6: bytes=32 time=0ms TTL=128

Ping statistics for 10.0.0.6:
Packets: sent=4, Recieved=4, Lost=0(0% loss),
approximate round trip times in milliseconds:
Minimum=0ms, Maximum=0ms, Average=0ms.

Hub ON switch OFF

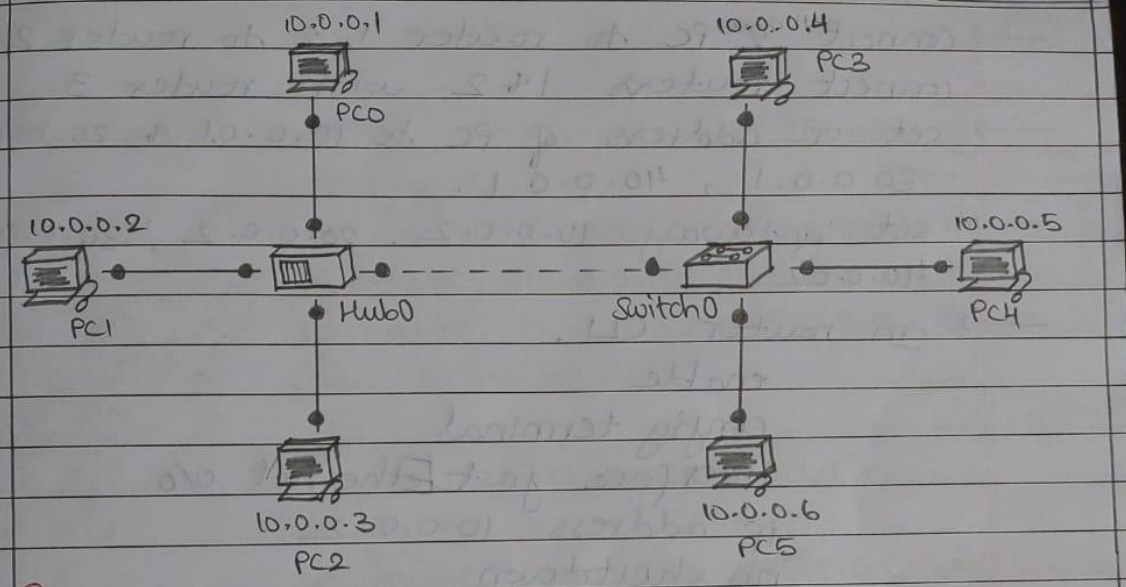
PC> ping 10.0.0.4
pinging 10.0.0.4 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 10.0.0.4:
Packets: sent=4, recieved=0, lost=4 (100% loss)

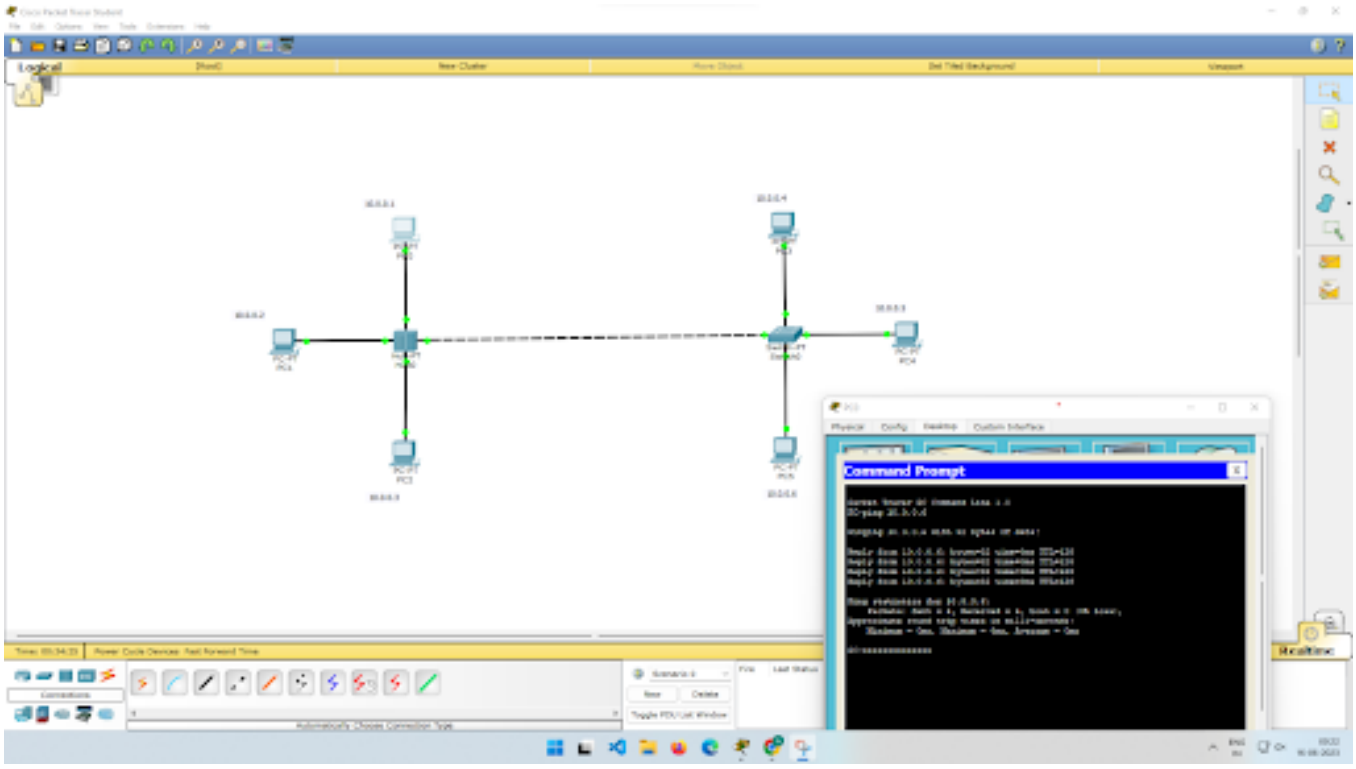
DATE: PAGE:

Topology:



16/6

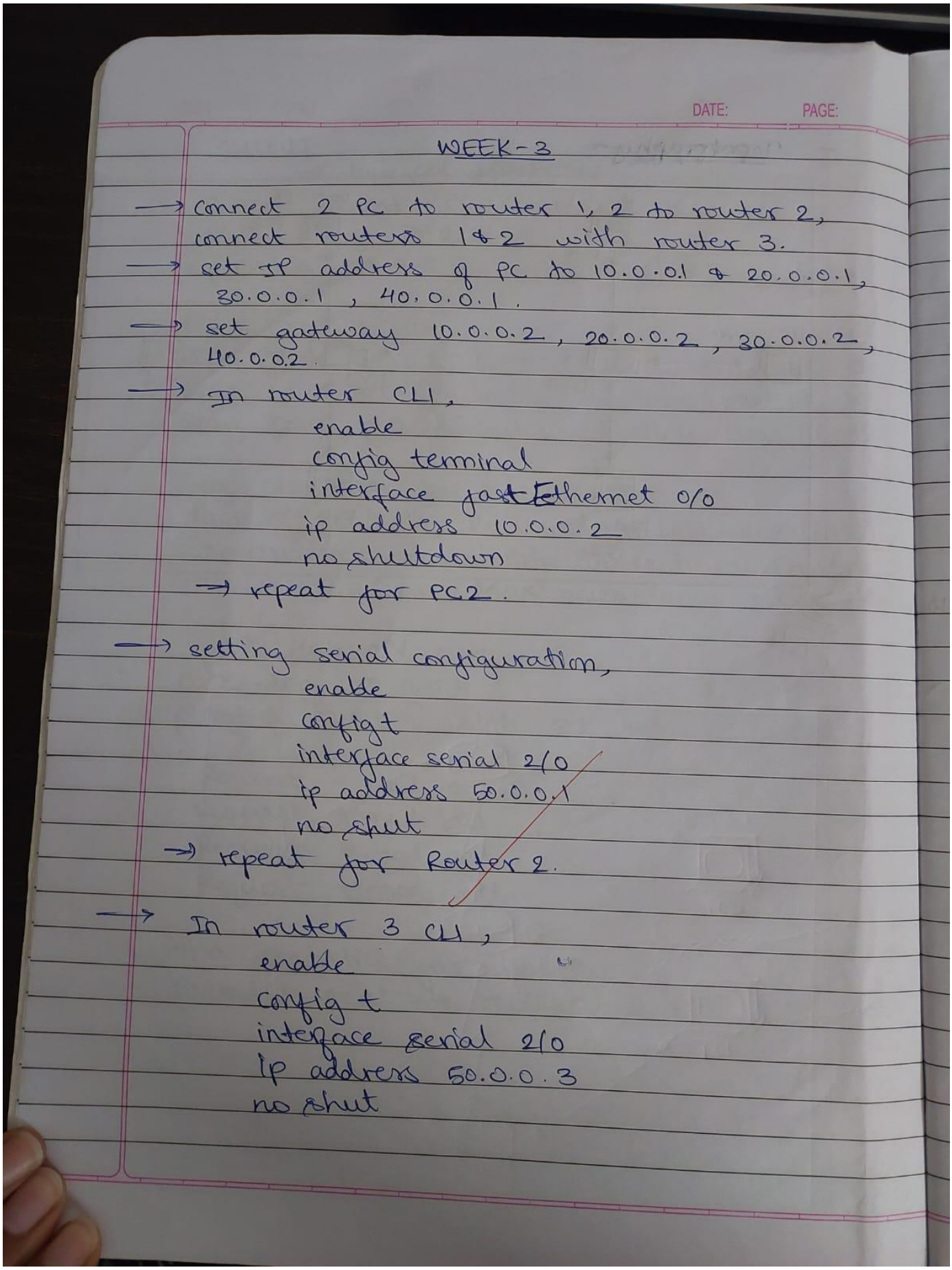
System Output-



LAB 3

Configure default route, static route to the Router.

Observation-



DATE: PAGE:

WEEK-3

→ Connect 2 PC to router 1, 2 to router 2, connect routers 1 & 2 with router 3.

→ set IP address of PC to 10.0.0.1 & 20.0.0.1, 30.0.0.1 , 40.0.0.1.

→ set gateway 10.0.0.2 , 20.0.0.2 , 30.0.0.2 , 40.0.0.2.

→ In router CLI,

enable
config terminal
interface ~~fast~~ Ethernet 0/0
ip address 10.0.0.2
no shutdown

→ repeat for PC2.

→ setting serial configuration,

enable
config t
interface serial 2/0
ip address 50.0.0.1
no shut

→ repeat for Router 2.

→ In router 3 CLI,

enable
config t
interface serial 2/0
ip address 50.0.0.3
no shut

ip (fire, status) short

DATE: PAGE:

interface serial 3/0
ip address 60.0.0.3
no shut

→ Static routing, (In router 1 CLI)

config t
ip route 30.0.0.0 255.0.0.0 50.0.0.3 (next hop / trace from source)
ip route 40.0.0.0 255.0.0.0 50.0.0.3
ip route 60.0.0.0 255.0.0.0 50.0.0.3
show ip route

done after reloading destination host unreachable, then initial show ip route

(In router 2 CLI)

config t
ip route 10.0.0.0 255.0.0.0 60.0.0.3
ip route 20.0.0.0 255.0.0.0 60.0.0.3
ip route 50.0.0.0 255.0.0.0 60.0.0.3

(In router 3 CLI)

config t
ip route 10.0.0.0 255.0.0.0 50.0.0.1
ip route 20.0.0.0 255.0.0.0 50.0.0.1
ip route 30.0.0.0 255.0.0.0 60.0.0.1
ip route 40.0.0.0 255.0.0.0 60.0.0.1

→ In command prompt of the PCs, we
ping 20.0.0.1 (from 10.0.0.1)

ping 2x
ping -t "3" ⇒ ping 30.0.0.1 (from 10.0.0.1)

record the output.

DATE: PAGE:

- OUTPUT -

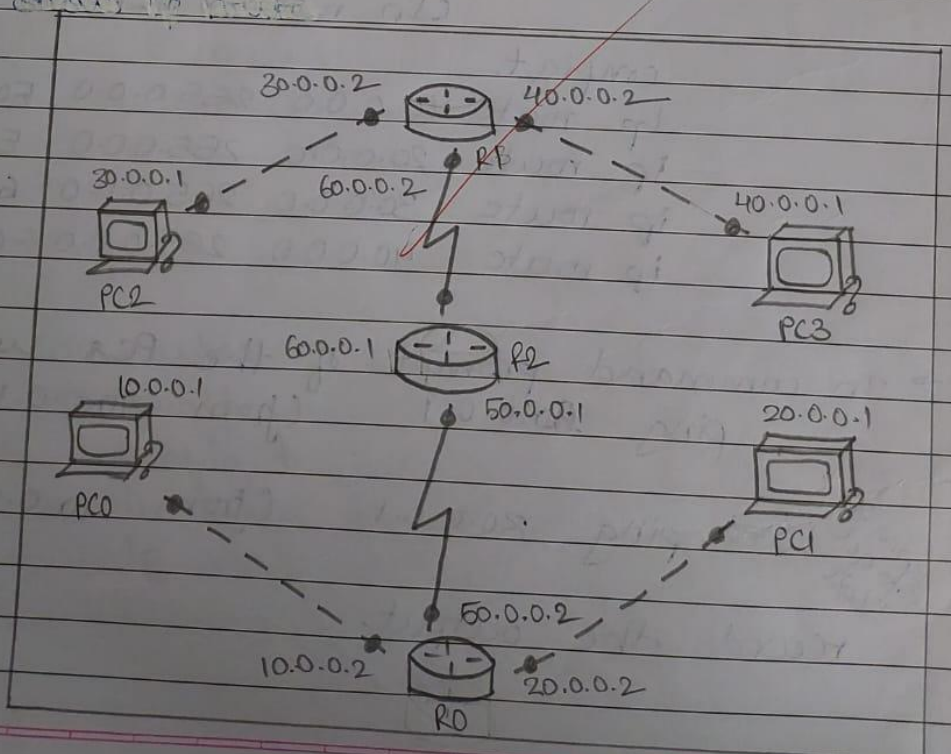
(ii) >ping 30.0.0.1

pinging 30.0.0.1 with 32 bytes of data:

Reply from 30.0.0.1: bytes=32 time=2ms TTL=125
Reply from 30.0.0.1: bytes=32 time=5ms TTL=125
Reply from 30.0.0.1: bytes=32 time=11ms TTL=125
Reply from 30.0.0.1: bytes=32 time=13ms TTL=125

Ping statistics for 30.0.0.1:
Packets: sent=4, recieved=4, Lost=0 (0% loss),
approximate round trip times in milli-seconds:
minimum=2ms, maximum=13ms, average=7ms

- Topology -



DATE: PAGE:

– OUTPUT –

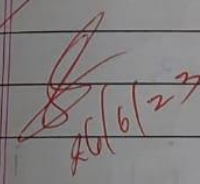
(i) (Initial)

> ping 40.0.0.1

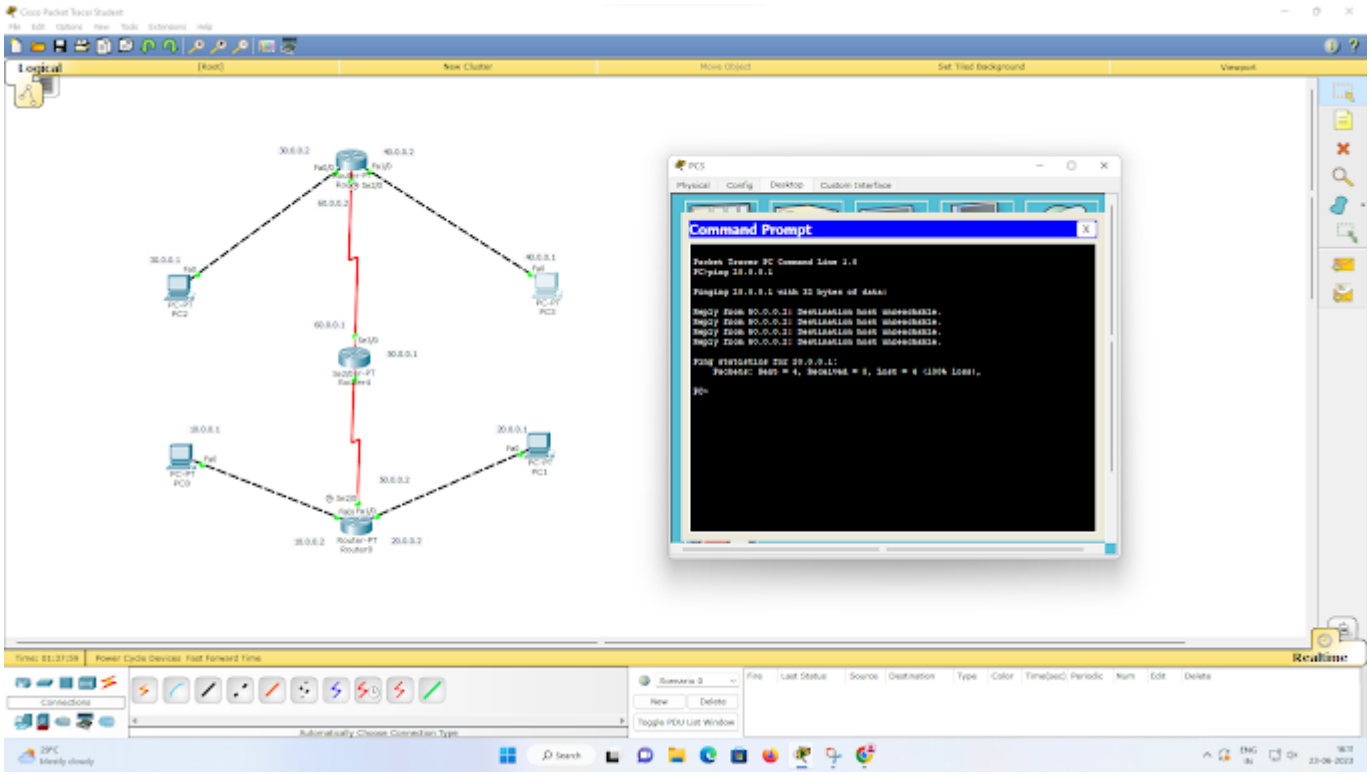
pinging 40.0.0.1 with 32 bytes of data:

Reply from 20.0.0.2: destination host unreachable.
Reply from 20.0.0.2: destination host unreachable
Reply from 20.0.0.2: destination host unreachable
Reply from 20.0.0.2: destination host unreachable.

Pinging statistics for 40.0.0.1:
Packets: sent = 4, recieved = 0, lost = 4 (100% loss)



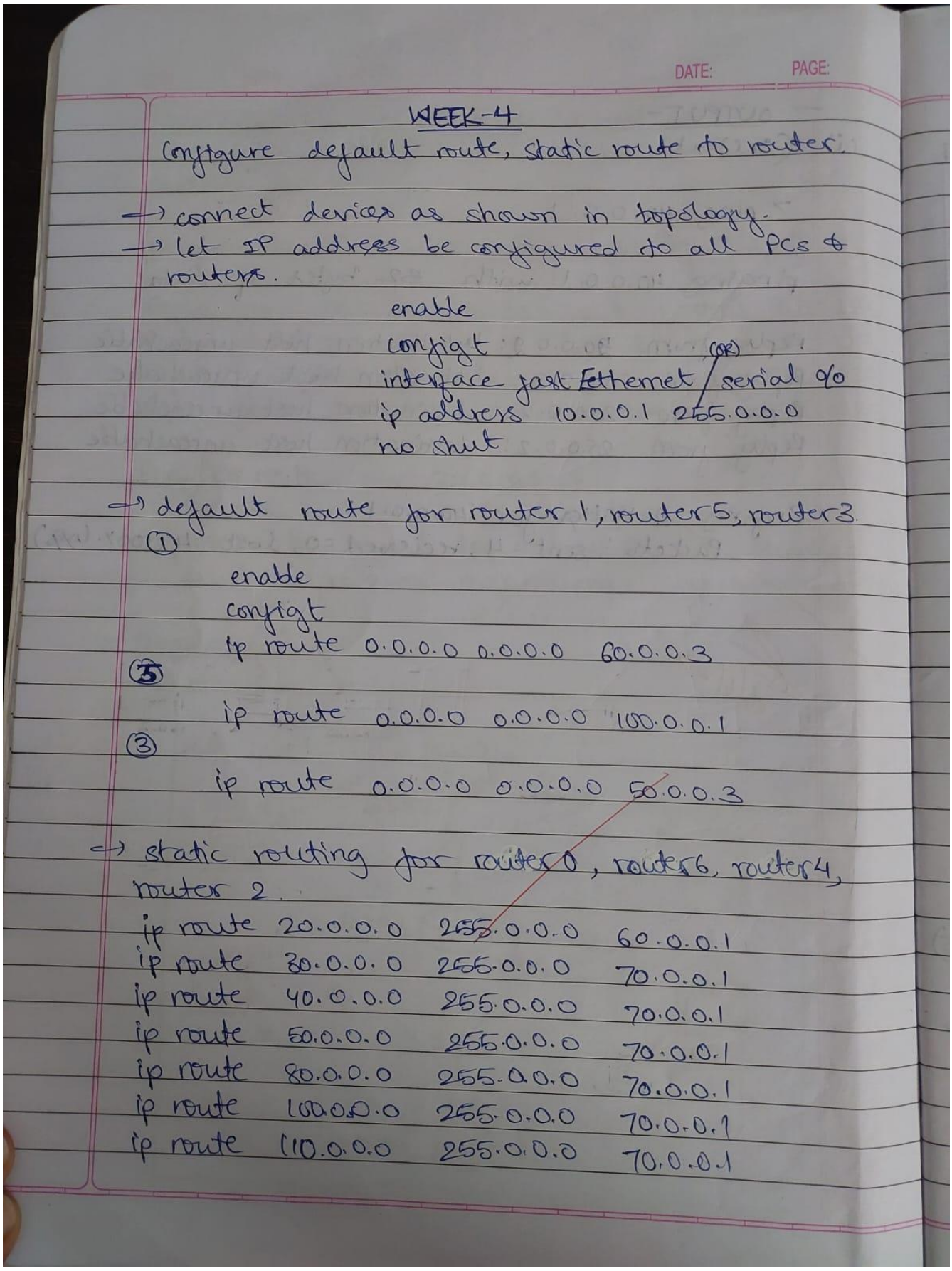
System Output-



LAB 4

Configure DHCP within a LAN and outside LAN

Observation-



DATE: PAGE:

WEEK-4

Configure default route, static route to router.

→ connect devices as shown in topology.
→ let IP address be configured to all PCs & routers.

```
enable
configt
interface fast Ethernet / serial 0/0
ip address 10.0.0.1 255.0.0.0
no shut
```

→ default route for router 1, router 5, router 3.

①
```
enable
configt
ip route 0.0.0.0 0.0.0.0 60.0.0.3
```
⑤
```
ip route 0.0.0.0 0.0.0.0 100.0.0.1
```
③
```
ip route 0.0.0.0 0.0.0.0 50.0.0.3
```

→ static routing for router 0, router 6, router 4, router 2.

```
ip route 20.0.0.0 255.0.0.0 60.0.0.1
ip route 30.0.0.0 255.0.0.0 70.0.0.1
ip route 40.0.0.0 255.0.0.0 70.0.0.1
ip route 50.0.0.0 255.0.0.0 70.0.0.1
ip route 80.0.0.0 255.0.0.0 70.0.0.1
ip route 100.0.0.0 255.0.0.0 70.0.0.1
ip route 110.0.0.0 255.0.0.0 70.0.0.1
```

:61

DATE: PAGE:

continue to do so for others.

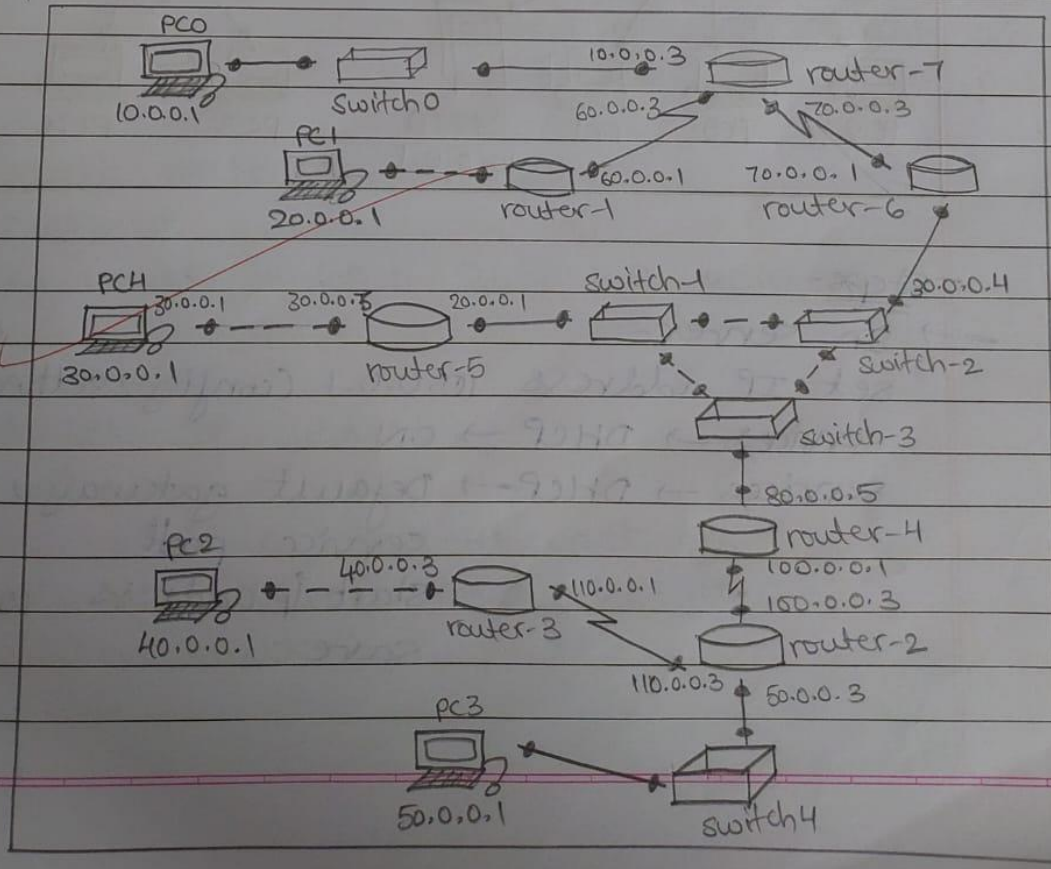
OUTPUT-

ping 20.0.0.1
pinging 20.0.0.1 with 32 bytes of data.

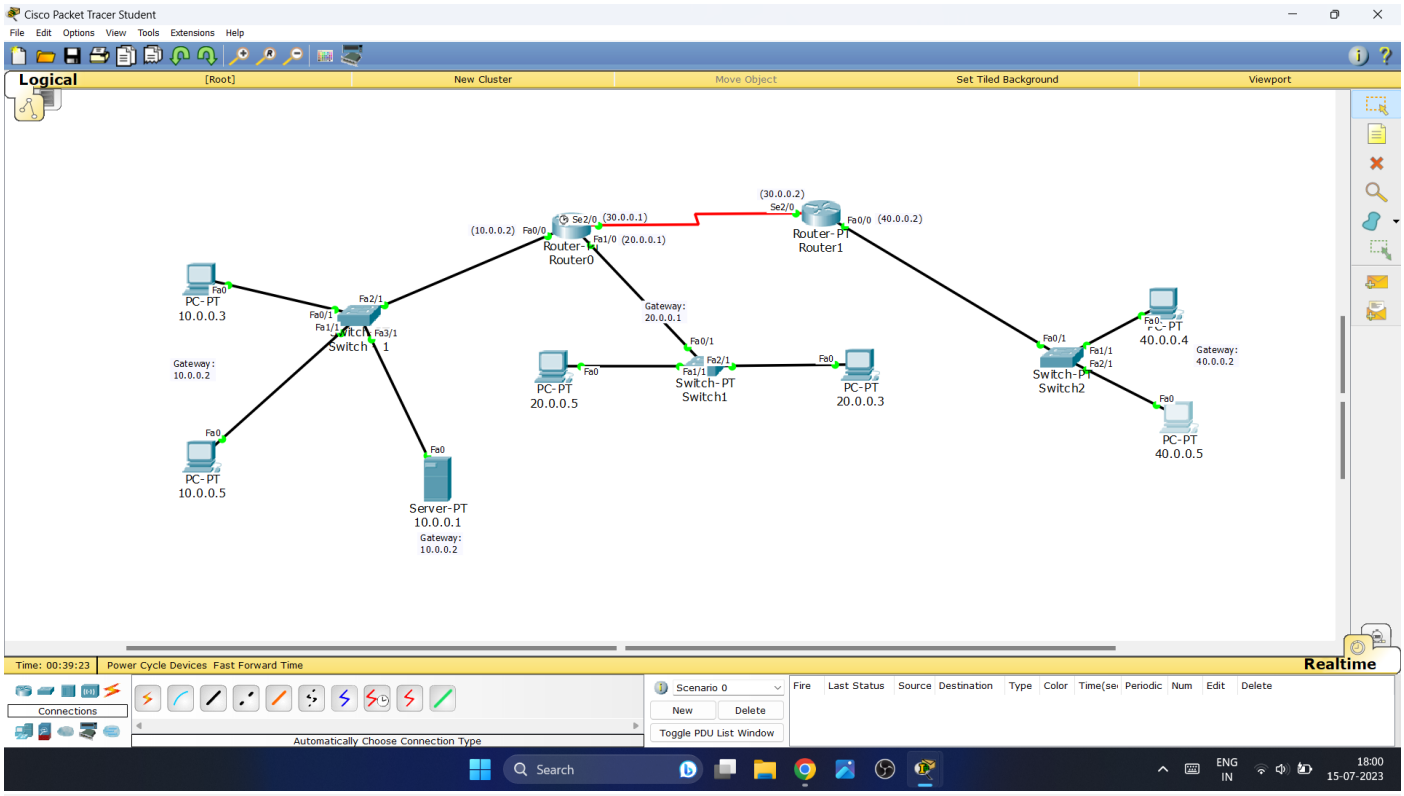
Request time out
Reply from 20.0.0.1: bytes = 32 time = 7ms TTL = 126
Reply from 20.0.0.1! bytes = 32 time = 7ms TTL = 126
Reply from 20.0.0.1! bytes = 32 time = 7ms TTL = 126

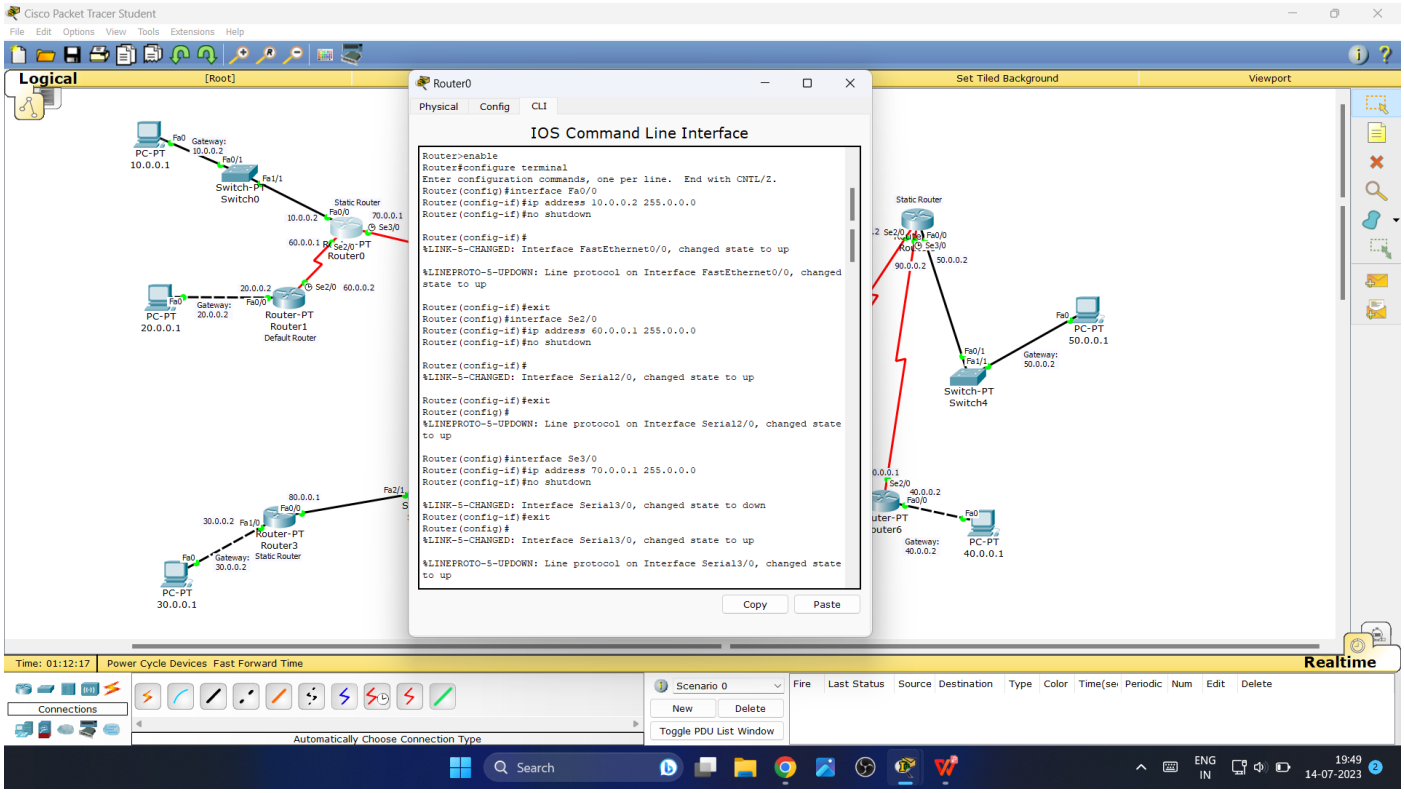
ping statistics for 20.0.0.1!
packets: sent = 4, recieved = 3, lost = 1 (25% loss)
approximate round trip times in milliseconds
minimum = 6ms, maximum = 7ms, average = 6ms

TOPOLOGY-



System Output-

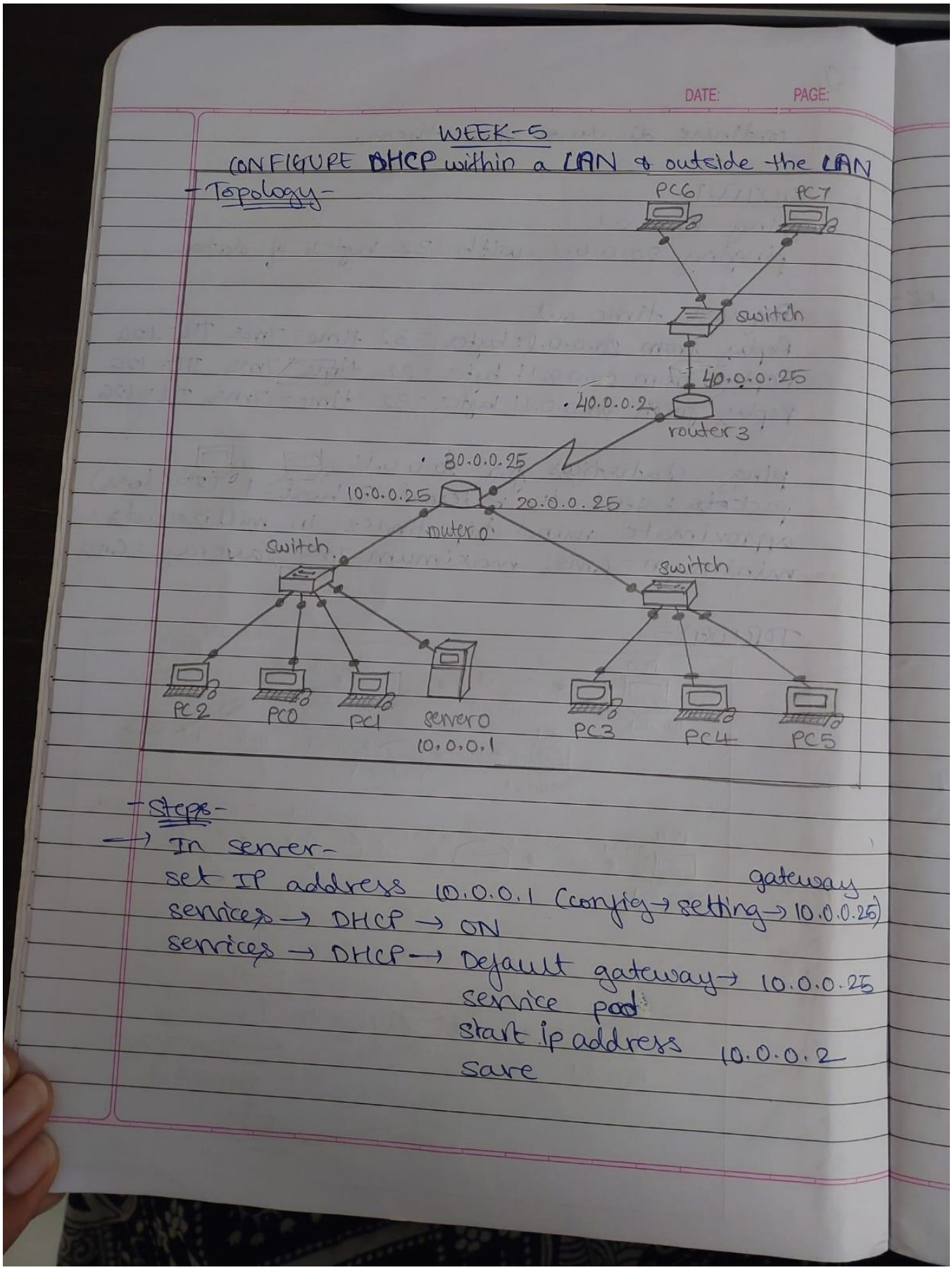




LAB 5

Configure Web Server, DNS within a LAN.

Observation-



DATE: PAGE:

⇒ service pool 1
gateway → 10.0.0.25
start IP address 20.0.0.25
add

⇒ service pool 2
gateway → 10.0.0.25
start IP address - 40.0.0.2
add

→ In router 0 - set 2 network ip addresses
⇒ config t
interface fastEthernet 4/0
ip address 10.0.0.25 255.0.0.0
no shut (∵ 20.0.0.25)
↓
config t
interface fastEthernet 0/0
ip helper-address 10.0.0.1
no shut

→ static route (for 40 ip)
⇒ config t
ip route 40.0.0.0 255.0.0.0 30.0.0.26

→ In router 1:
* set IP address
⇒ config t
interface fastEthernet 0/0
Ip address 40.0.0.26 255.0.0.0
no shut

DATE: PAGE:

⇒ config t
interface serial 2/0
Ip address 30.0.0.26 255.0.0.0
no shut

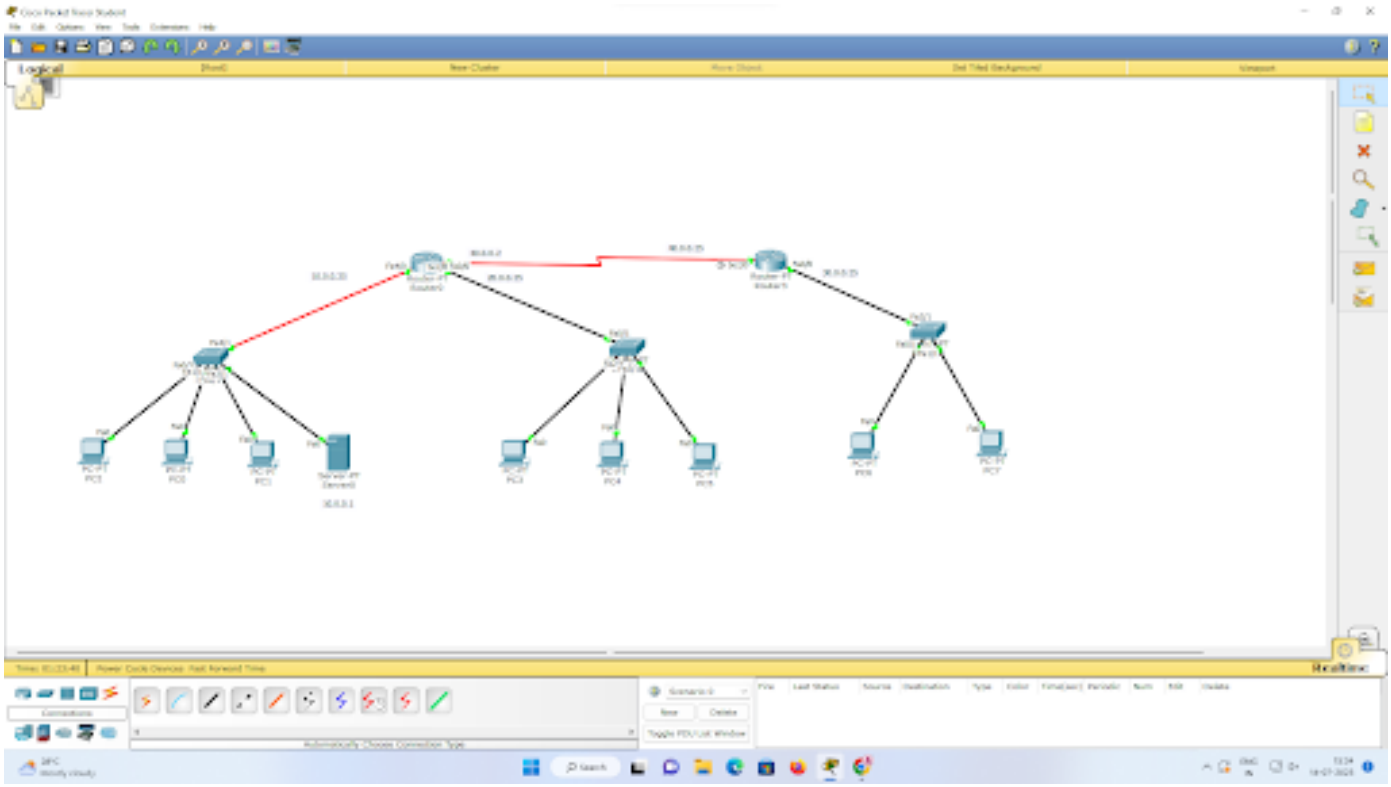
→ Static routing for 10 & 20 network in router3
⇒ config t
ip route 10.0.0.0 255.0.0.0 30.00.25
ip route 20.0.0.0 255.0.0.0 30.0.0.25
no shut

→ setting helper address
⇒ config t
interface gastEthernet 0/0
Ip helper-address 10.0.0.1
no shut

- Output - (In any PC)

Desktop → IP configuration → DHCP
(Dynamic IP address is assigned to all PCs by server).

System Output-



LAB 6
Configure RIP routing Protocol in Routers

Observation-

DATE: PAGE:

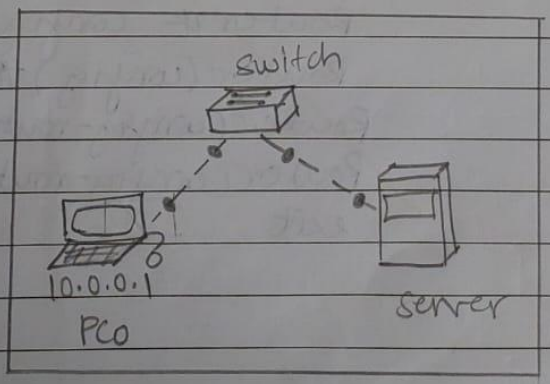
WEEK-6
DNS

1) Create a topology (1PC, 1S, 1GSer.)
2) configuration of PC (IP address, gateway)
   configuration of server (IP address) → CLI / web browser
3) Server →

1. -Procedure-

→ Create a topology by placing a PC, a server & a switch on the workspace.
→ configure PC & server (IP address + gateway (PC))
→ open web browser of PC to set IP address of server.
→ configure DNS of server with name (website name) & URL (IP address)
→ Edit index.html to display USN & Name.

-Topology-



Output- Prakhyati

Router Interface Protocol DATE: PAGE:

11 = Procedure-

→ create a topology with 2PC, 3 routers
→ configure PC with IP address & gateway
→ configure routers-
for router 0:
enable
config terminal
interface fastethernet 0/0
ip address 10.0.0.1 255.0.0.0
interface serial 2/0
ip address 10.0.0.2
encapsulation ppp ← all routers
only where present → clock rate 64000
no shut
exit

→ configure all routers-
for router 0:
Router enable
Router# show ip route
Router# config t
Router(config)# router rip
Router(config-router)# network 10.0.0.0
Router(config-router)# network 20.0.0.0
exit

DATE: PAGE:

- output -

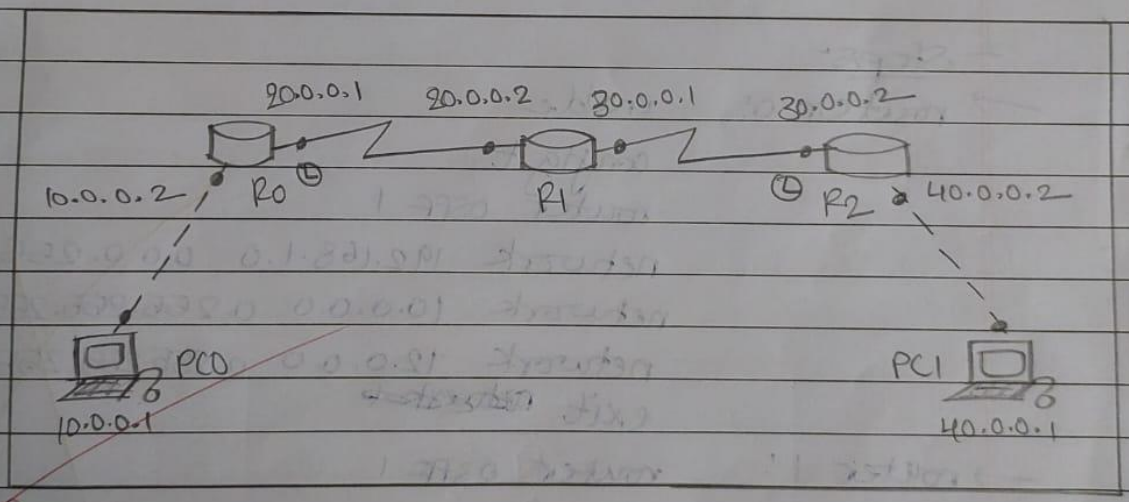
In PC0,

⇒ ping 40.0.0.1

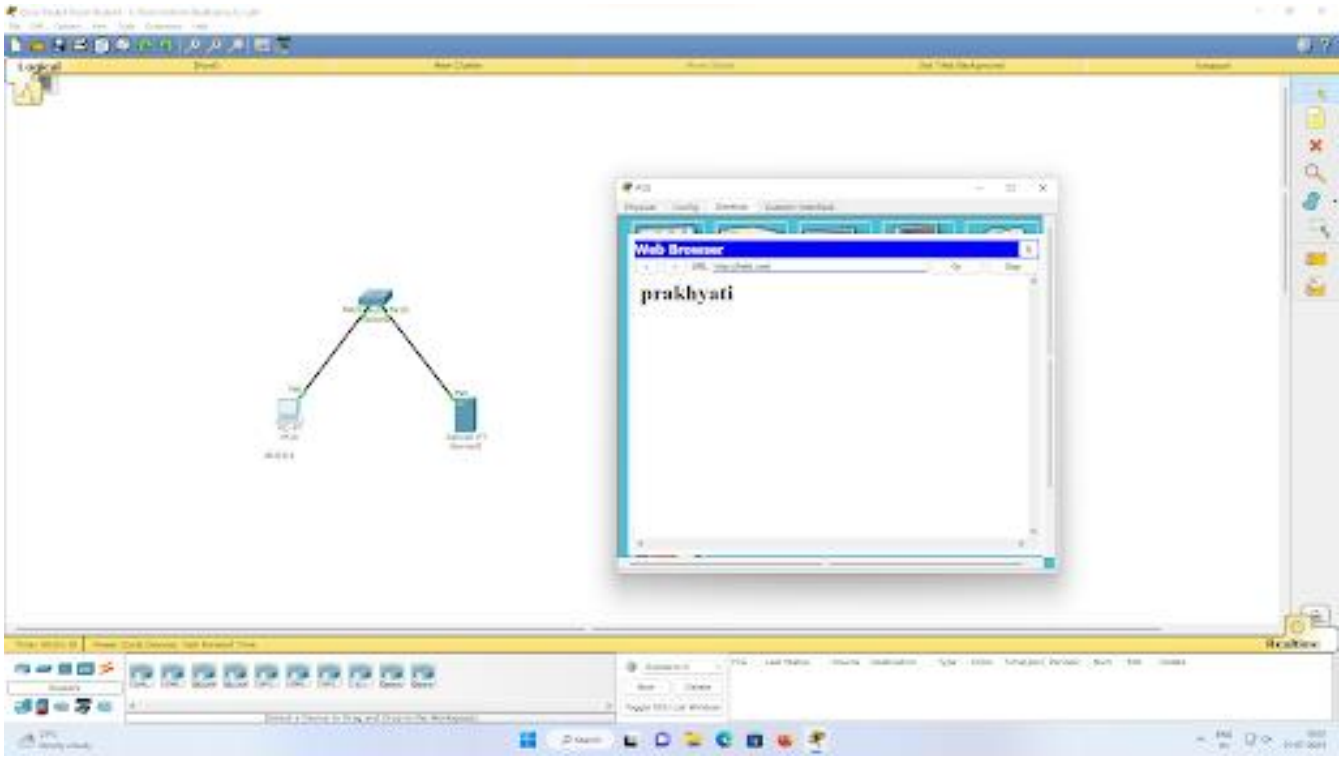
pinging 40.0.0.1 with 32 bytes of data:
Request timed out.
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Ping statistics for 40.0.0.1:
Packets: sent=4, Recieved=3, Lost=1 (25% loss)

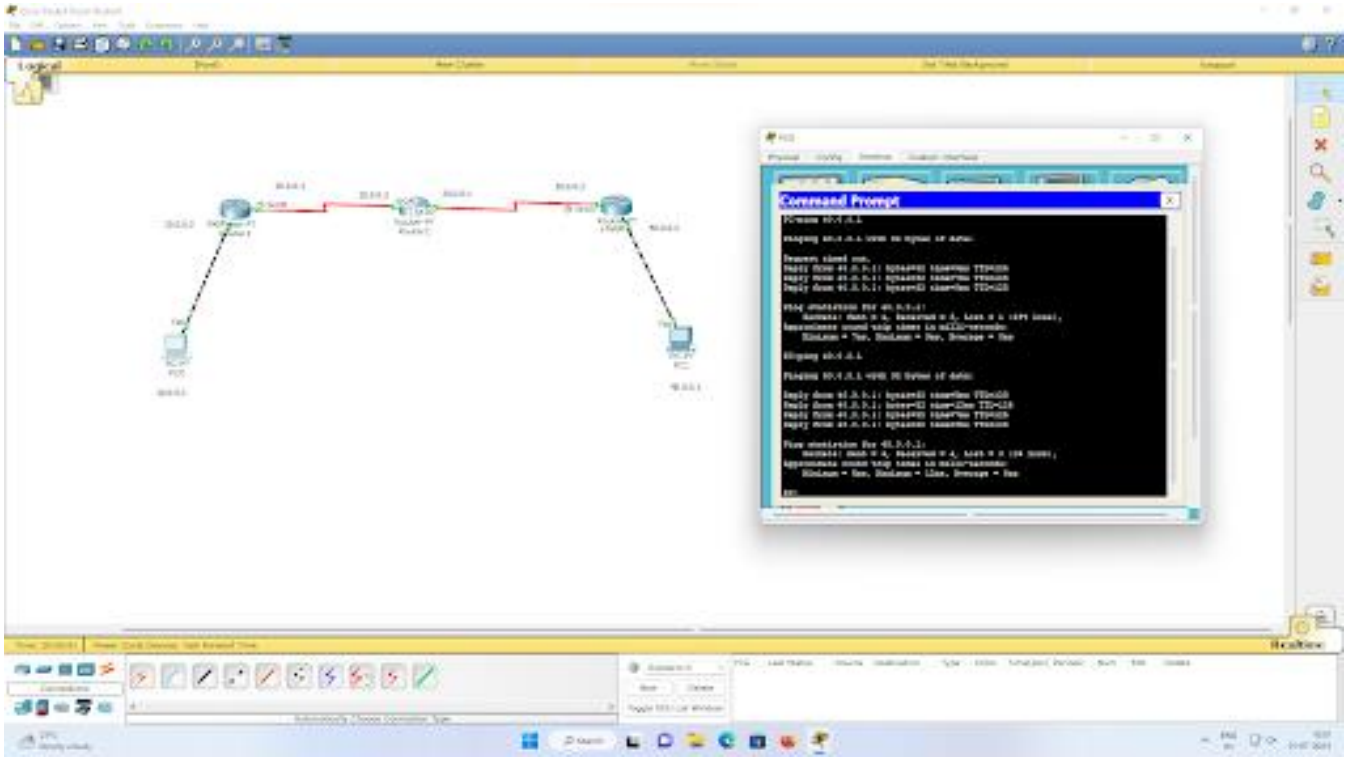
⇒ ping 40.0.0.1 (for no loss, ping again)

- Topology -



System Output-





LAB 7
Configure OSPF routing protocol

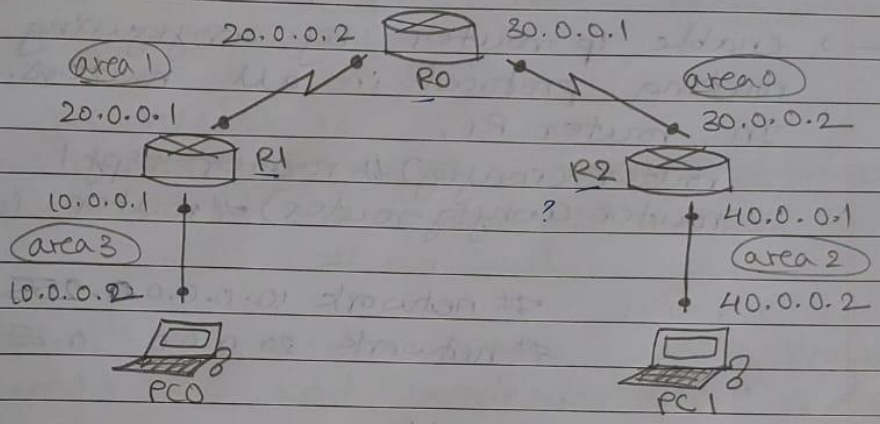
Observation-

WEEK-7

DATE: PAGE:

(1) Config. OSPF routing protocol & connect areas

– Topology:



– Procedure:

→ create topology w/ 2PC & 3 routers

→ config. ip address for PC as 10.0.0.2 & 40.0.0.2 :

→ config. routers w/ ip address for all interfaces.

→ for all se ports of routers, config using cmd "encapsulation ppp" & give "clock rate 64000" command at ports having clock symbol.

⇒ for Router 2

router (config) # interface se 2/0
router (config-if) # ip encapsulation ppp
no shut
exit

DATE: PAGE:

router(config)# interface se 3/0
router(config-if)# encapsulation ppp
clock rate 64000
no shut

→ enable ip router by configuring OSPF routing protocol in all routers.
In router R1,
router(config)# router ospf 1
router(config-router)# router-id 1.1.1.1
area 3
network 10.0.0.0 0.255.255.255
network 20.0.0.0 0.255.255.255
area 1
exit
We config 4 to routers 2 & 3.

→ check routing table of R1
router# show ip route
codes C-connected, S-static, R-RIP, M-Mobile, B-BGP, O-OSPF, IA-OSPF, inter area N1-OSPF NSS A extend type 1.

gateway of last resort is not set
C 10.0.0.0/8 is directly connected, ja 2/0
C 20.0.0.0/8 is directly connected, se 2/0
OIA 40.0.0.0/8 via 20.0.0.2, 00:04:23, se 2/0
OIA 30.0.0.0/8 via 20.0.0.2, 00:07:29, se 2/0
R2 knows area 0, network 20.0.0.0 connected to R2 from R1, so R1 learns through this network.

DATE: PAGE:

→ there must be 1 interface up to keep OSPF process up. So it's better to config. loopback address to routers. Its virtual interface, never goes down once we config. it.

for router 1:

router config-if)# interface loopback 0
ip add 172.16.1.252 255.255.0.0
no shut

do ||le for router 2 & 3. using these cmds we add loopback address to the routers.

→ If we check the routing table from R3,

R3 # show ip route

codes: C - connected, S - static, O - OSPF, IA - OSPF

gateway of last resort is not set

O IA 20.0.0.0/8 [110/128] via 30.0.0.1, 00:18:58, se 2/0

C 40.0.0.0/8 is directly connected; fa 0/0

C 30.0.0.0/8 is directly connected; se 0/0

R3 (router 3) doesn't know about area 3, so, we will create virtual link b/w R1 & R3.

→ we have to create virtual link b/w R1 & R2; with this we create a virtual link to connect area 3 to 0.

DATE: PAGE:

In router R1:
router (config) # router OSPF 1
router (config-router) # area 1 virtual link 2.2.2.2

∵ R2 of virtual link 1.1.1.1
exit.

→ R2 & R3 get updates about area 3,
check routing table for R3.

router # show ip route
codes: C-connected, O-OSPF, IA-OSPF,
gateway of last resort is not set.

O/A 20.0.0.0/8 [110/128] via 30.0.0.1, 00:01:56, se2/0
C 40.0.0.0/8 is directly connected.
O/A 10.0.0.0/8 [110/128] via 30.0.0.1, 00:01:56, se2/0
C 30.0.0.0/8 is directly connected, se 2/0

→ Now, ping 10.0.0.2 to 40.0.0.2 for o/p.

- OUTPUT-
> ping 40.0.0.2 with 32 bytes of data:

reply from 40.0.0.2 : bytes=32 time=2ms TTL=125
reply from 40.0.0.2 : bytes=32 time=2ms TTL=125
reply from 40.0.0.2 : bytes=32 time=2ms TTL=125
reply from 40.0.0.2 : bytes=32 time=2ms TTL=125

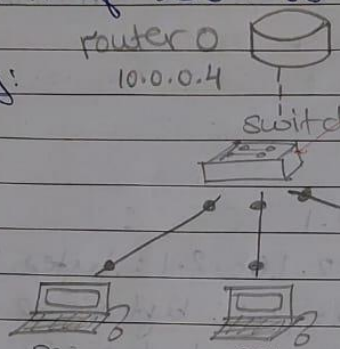
ping statistics for 40.0.0.2:
packets: sent=4, recieved=4, lost=0

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

DATE: PAGE:

(I) Construct a LAN & understand the concept & operation of address resolution protocol (ARP)

- Topology:



- Steps:

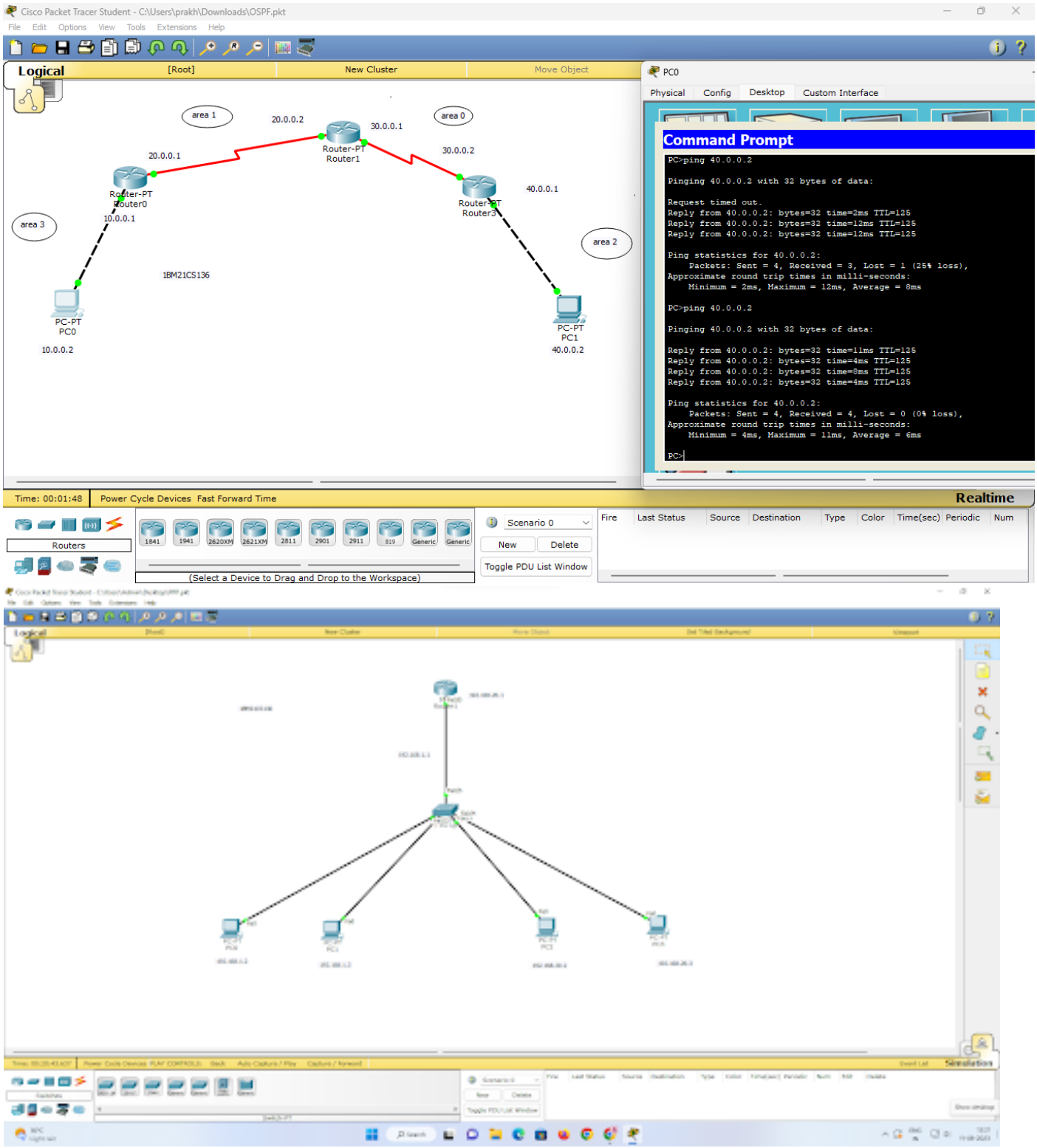
→ configure IP: PC0 (CMD)

```
arp -a
ping 10.0.0.2
arp -a
arp -d
```

- OUTPUT:

```
arp -a (no ARP entries found)
→ ping 10.0.0.2
    reply from 10.0.0.2 bytes = 32 time 0ms TTL = 128
    reply from 10.0.0.2 bytes = 32 time 0ms TTL = 128
    reply from 10.0.0.2 bytes = 32 time 0ms TTL = 128
→ arp -a
  internet address    physical address    type
    10.0.0.2          0002.1615.9820      dynamic
```

System Output-



LAB 8

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

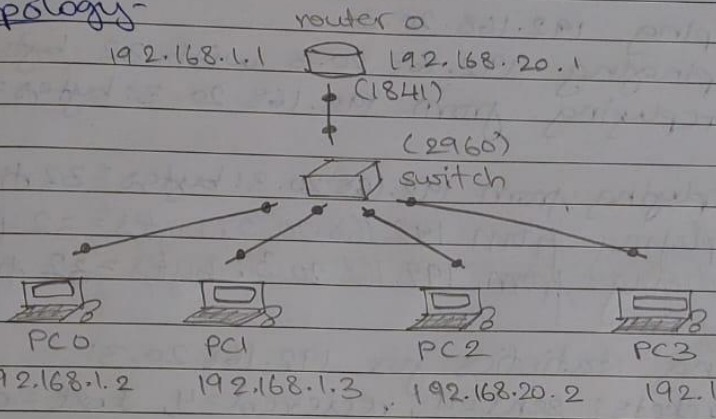
Observation-

11/8/23

VLAN

(I)

- Topology-



- Steps:

→ In switch → VLAN database → configure VLAN number & VLAN name

fastethernet 0/5 → In drop down, select TRUNK

fast ethernet 0/4 → configure VLAN

fast ethernet 0/3 → configure VLAN

→ In router

VLAN database → configure VLAN number & VLAN name

CLI of router:

config t

interface fastethernet 0/0.1

encapsulation dot1q 20

ip address 192.168.20.1 255.255.255.0

no shut

exit

DATE: PAGE:

- OUTPUT-

ping 192.168.20.3
pinging 192.168.20.3 with 32 bytes of data!
replying from 192.168.20.3: bytes=32 time=0ms TTL=127

replying from 192.168.20.3: bytes=32 time=0ms TTL=127
replying from 192.168.20.3: bytes=32 time=0ms TTL=127
replying from 192.168.20.3: bytes=32 time=0ms TTL=127

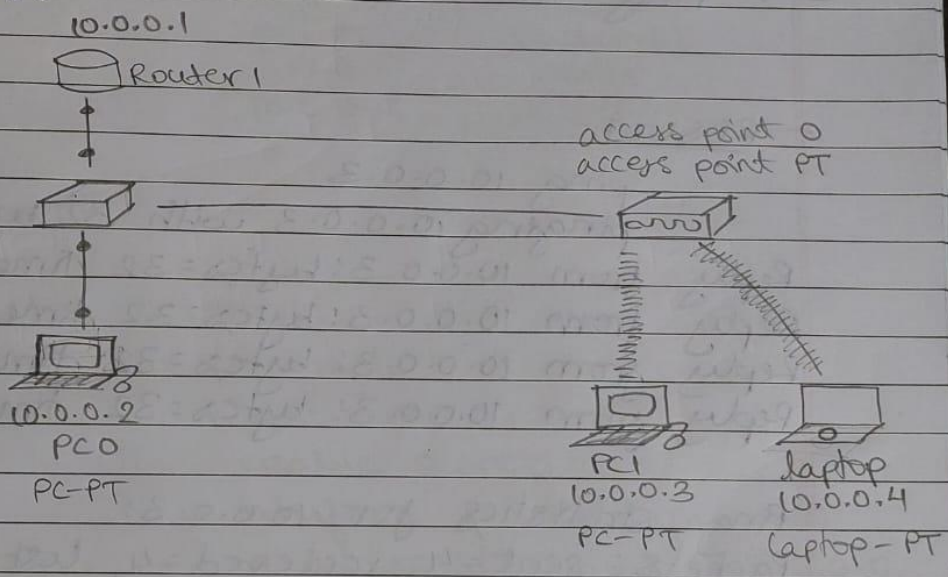
ping statistics for 192.168.20.3:
packets: sent=4, recieved=4, lost=0 (0% loss)

To construct a WLAN and make the nodes communicate wirelessly.

DATE: PAGE:

(VI) WLAN

— Topology —



— steps —

→ construct the topology.

→ set ip address & gateway of PC.

→ for access point 0,

⇒ In port 1

SSID ⇒ WLAN

WLAN key ⇒ 1234567890

→ for PC 1,

switch off, drag the existing PT-HOST-NM-1AM to the components list & drag WMP300N to empty port. Then switch on device.

In wireless 0,

SSID ⇒ WLAN

WEN key ⇒ 1234567890

DATE: PAGE:

IP address ⇒ 10.0.0.3
gateway ⇒ 10.0.0.1

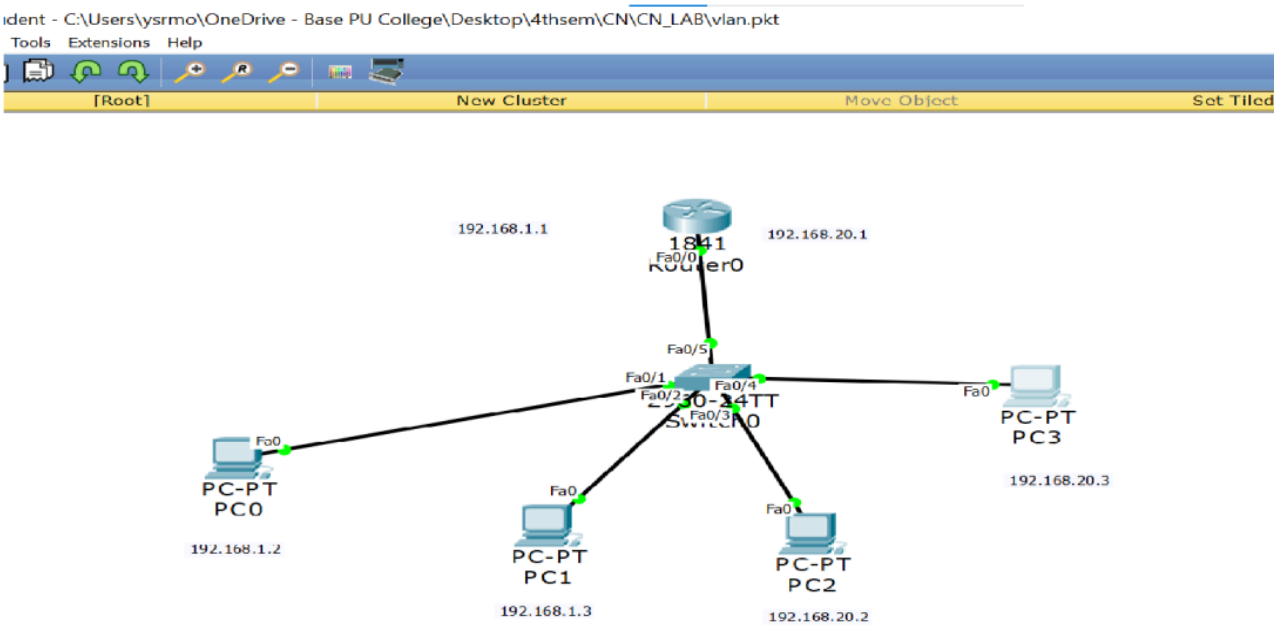
→ Repeat the same to laptop.
→ Ping the device.

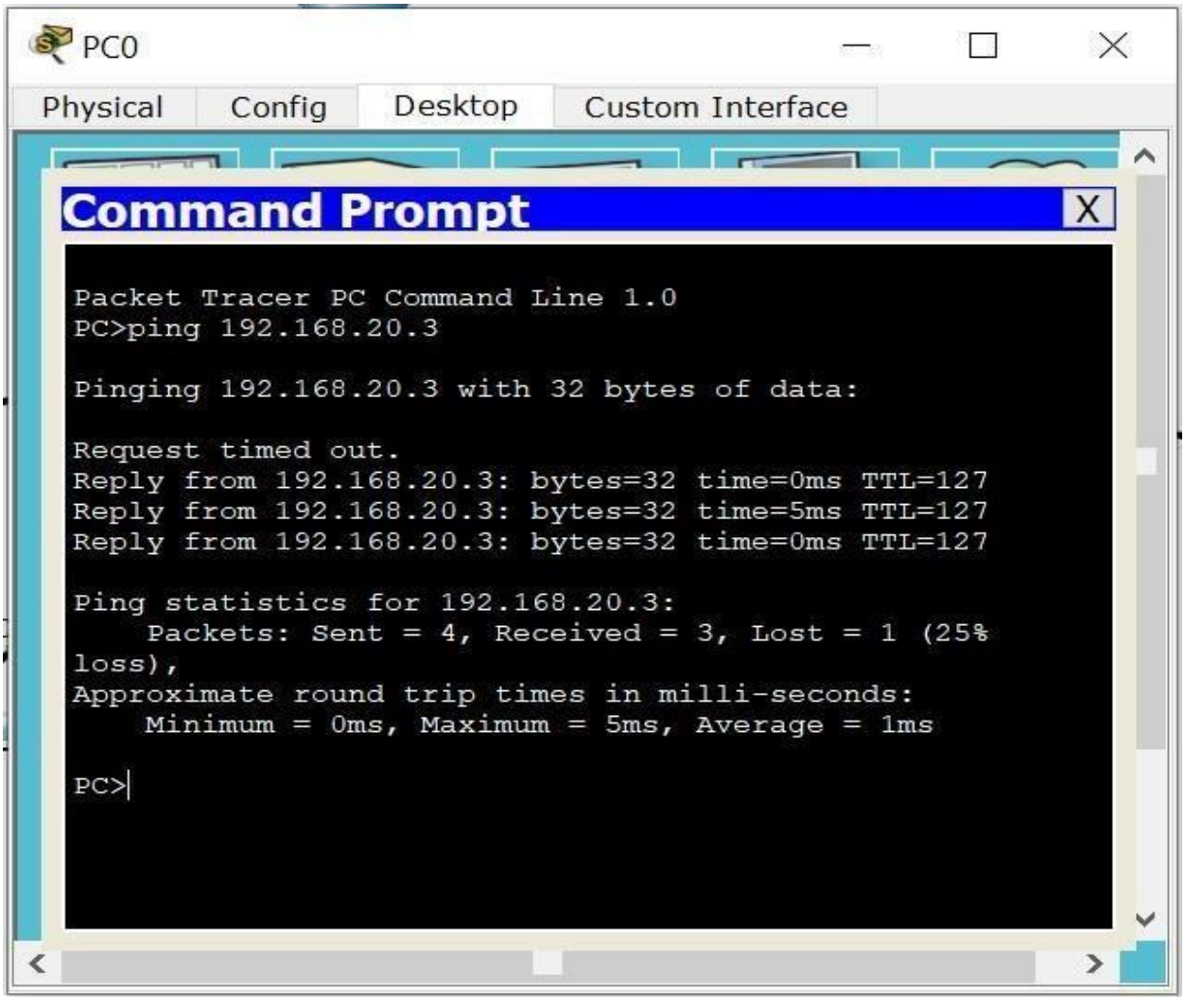
- RESULT -

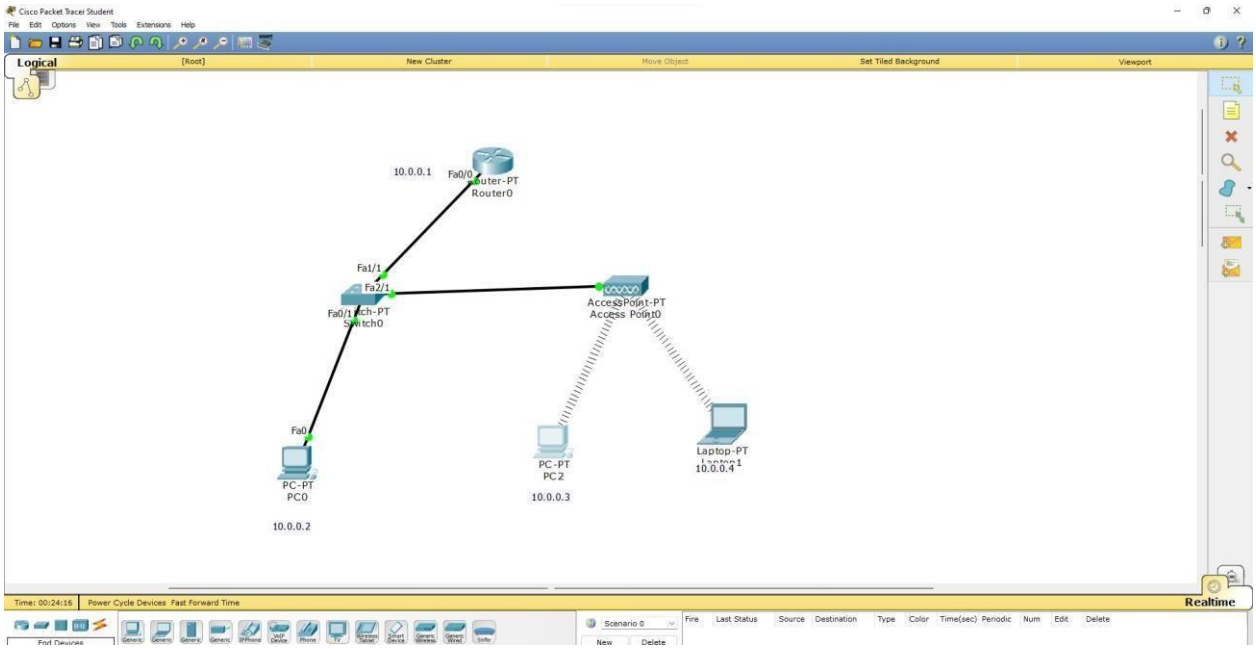
ping 10.0.0.3
pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3: bytes = 32 time = 15ms TTL = 128
Reply from 10.0.0.3: bytes = 32 time = 15ms TTL = 128
Reply from 10.0.0.3: bytes = 32 time = 15ms TTL = 128
Reply from 10.0.0.3: bytes = 32 time = 15ms TTL = 128

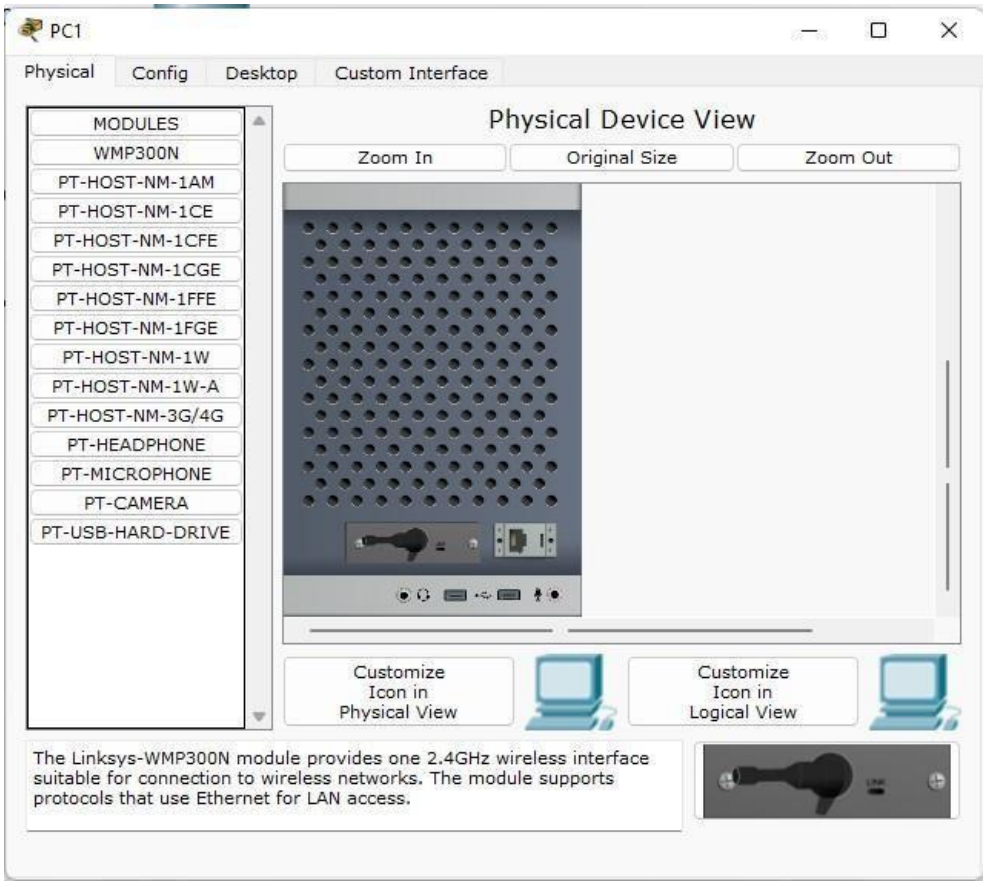
Ping statistics for 10.0.0.3:
Packets: sent = 4, recieved = 4, lost = 0
approximate round trip times in ms:
minimum = 6ms, maximum = 15ms, average = 10ms

System Output-






Cisco Packet Tracer Student
File Edit Options View Tools Extensions Help
Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
10.0.0.1
Fa0/0
Router-PT
Router0
Fa1/1
Fa2/1
Fa0/1
Switch-PT
Switch0
Access Point-PT
Access Point0
Fa0
PC-PT
PC0
10.0.0.2
PC-PT
PC2
10.0.0.3
Laptop-PT
10.0.0.4
Time: 00:24:16
Power Cycle Devices Fast Forward Time
Realtime


PC1
Physical
Config
Desktop
Custom Interface
MODULES
WMP300N
PT-HOST-NM-1AM
PT-HOST-NM-1CE
PT-HOST-NM-1CFE
PT-HOST-NM-1CGE
PT-HOST-NM-1FFE
PT-HOST-NM-1FGE
PT-HOST-NM-1W
PT-HOST-NM-1W-A
PT-HOST-NM-3G/4G
PT-HEADPHONE
PT-MICROPHONE
PT-CAMERA
PT-USB-HARD-DRIVE
Physical Device View
Zoom In
Original Size
Zoom Out
Customize Icon in Physical View
Customize Icon in Logical View
The Linksys-WMP300N module provides one 2.4GHz wireless interface suitable for connection to wireless networks. The module supports protocols that use Ethernet for LAN access.

LAB 9

Demonstrate the TTL/ Life of a Packet.

Observation-

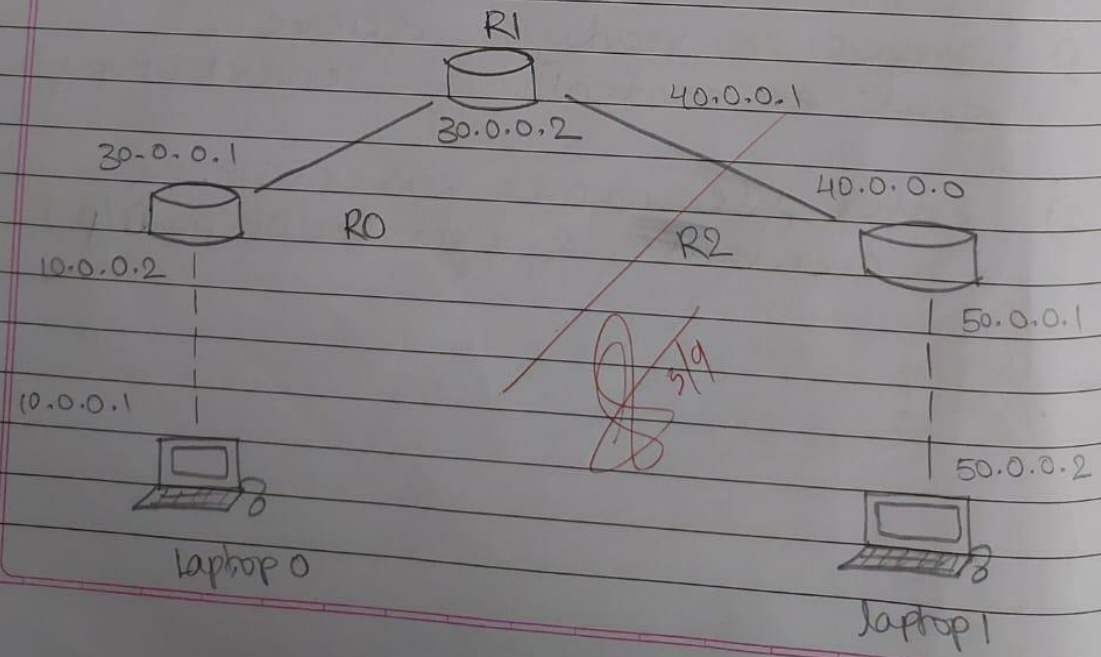
DATE: PAGE:

- Demonstrate TTL / life of a packet.

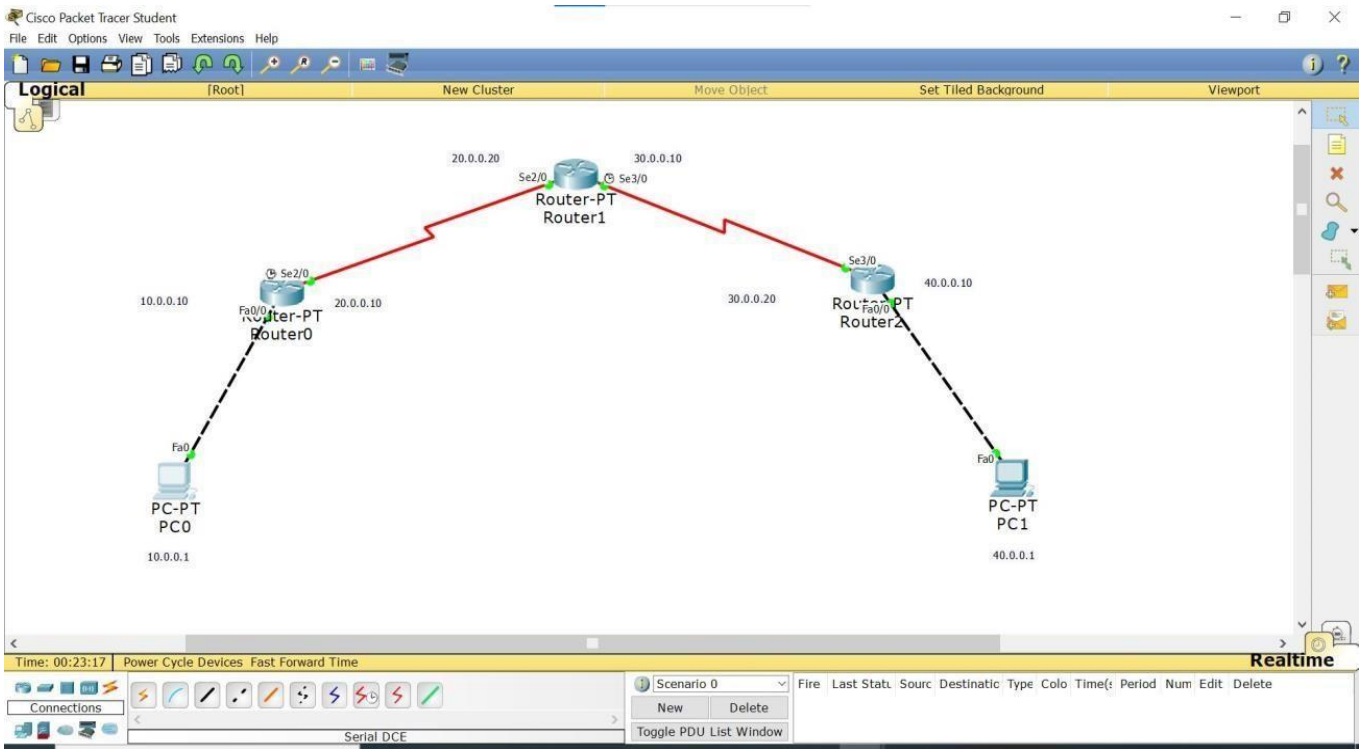
procedure:

→ create topology.
→ configure IP address of PC as 10.0.0.1 & 50.0.0.1
→ configure IP address of router
→ configure router using default / static routing.
→ simulation mode, send a simple PDU from 1 PC to another.
→ use capture button to capture every transfer.
→ click on PDU during every transfer to see the inbound & PDU details.

Topology:



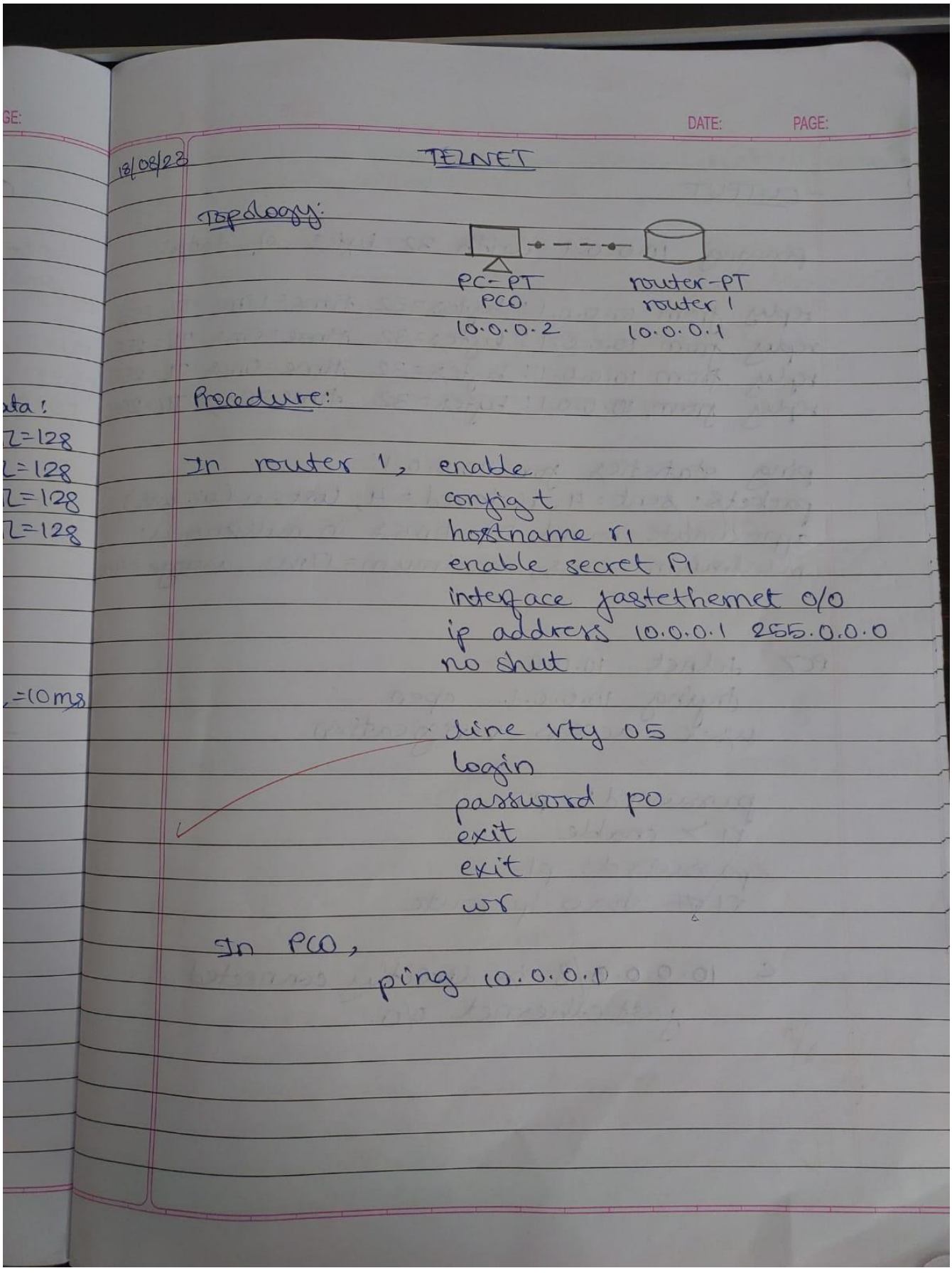
System Output-



LAB 10

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

Observation-



18/08/23

TELNET

Topology:

PC-PT
PC0
10.0.0.2

router-PT
router 1
10.0.0.1

Procedure:

In router 1, enable
config t
hostname r1
enable secret P1
interface fastethernet 0/0
ip address 10.0.0.1 255.0.0.0
no shut

line vty 05
login
password p0
exit
exit
wr

In PC0,
ping 10.0.0.1

DATE: PAGE:

-OUTPUT-

pinging 10.0.0.1 with 32 bytes of data!

reply from 10.0.0.1! bytes=32 time=17ms TTL=255
reply from 10.0.0.1! bytes=32 time=0ms TTL=255
reply from 10.0.0.1! bytes=32 time=0ms TTL=255
reply from 10.0.0.1! bytes=32 time=0ms TTL=255

ping statistics for 10.0.0.1!
packets: sent=4, recieved=4, lost=0 (0% loss),
approximate round trip times in milliseconds!
minimum=0ms, maximum=17ms, average=4ms

PC7 telnet 10.0.0.1
trying 10.0.0.1... open
user access verification

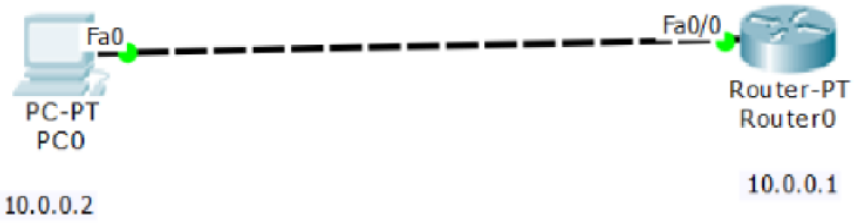
password: p0
r1> enable
password: p1
r1# show ip route

C 10.0.0.0/8 is directly connected,
fastethernet 0/0.

1/9

15

System Output-



PC0

Physical | Config | Desktop | Custom Interface

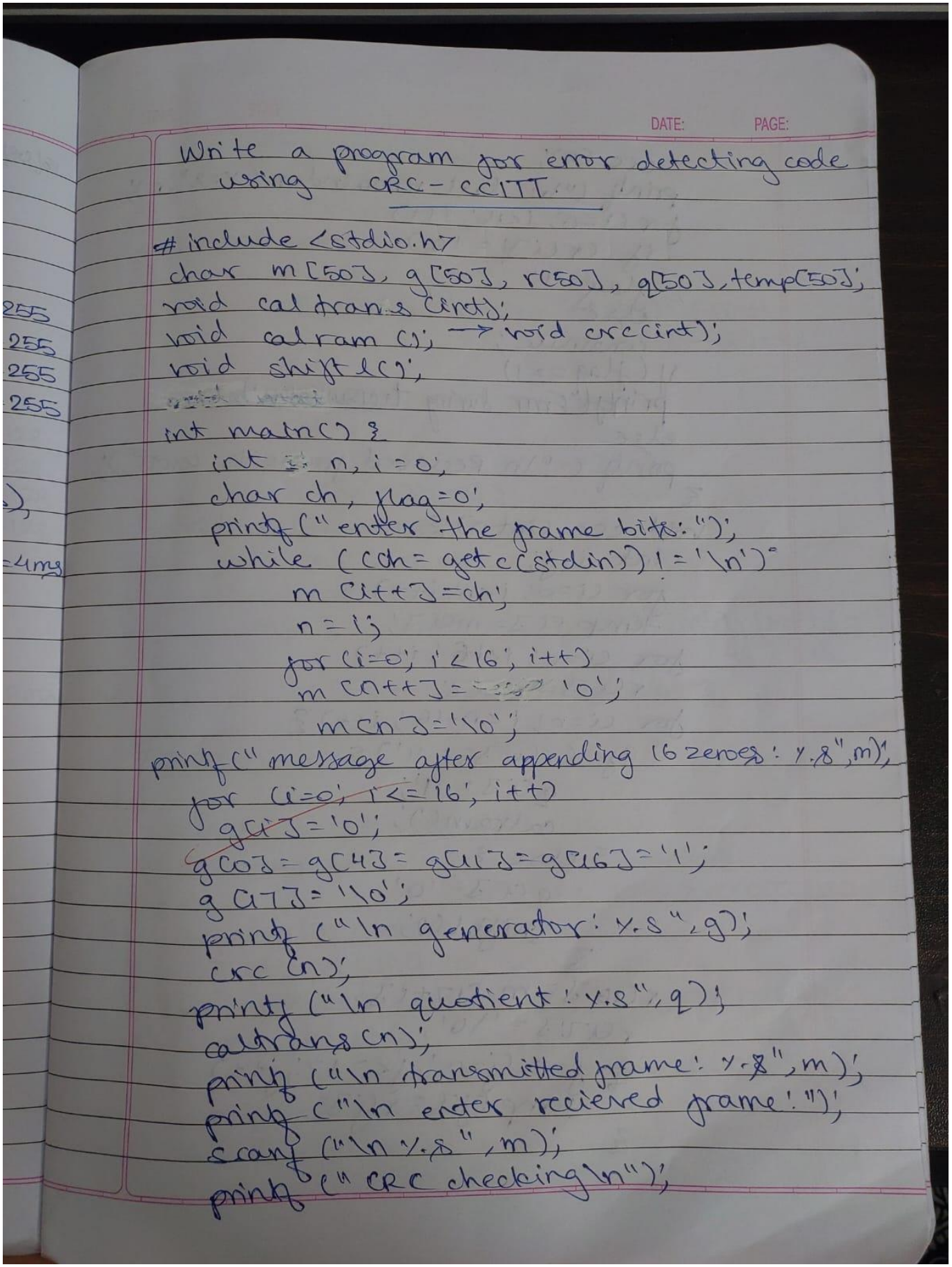
Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=1ms TTL=255
Reply from 10.0.0.1: bytes=32 time=0ms TTL=255
Reply from 10.0.0.1: bytes=32 time=0ms TTL=255
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),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open


User Access Verification

Password:
% Password:  timeout expired!

[Connection to 10.0.0.1 closed by foreign host]
PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open


User Access Verification

Password:
Password:
Password:

[Connection to 10.0.0.1 closed by foreign host]
PC>telnet 10.0.0.1
Trying 10.0.0.1 ...Open


User Access Verification

Password:
r1>enable
Password:
r1#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
r1#
```

LAB 11

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

Observation-



Write a program for error detecting code using CRC - CCITT.

```
#include<stdio.h>
char m[50], g[50], r[50], q[50], temp[50];
void caltrans(int);
void calram();  -> void crc(int);
void shift1();

int main(){
    int n, i=0;
    char ch, flag=0;
    printf("enter the frame bits: ");
    while((ch=getc(stdin))!='\n')
        m[i++]=ch;
        n=i;
        for(i=0; i<16; i++)
        m[n++]='0';
        m[n]='\0';
printf("message after appending 16 zeroes: %s",m);
    for(i=0; i<=16; i++)
    g[i]='0';
g[0]=g[4]=g[11]=g[16]='1';
g[17]='\0';
printf("\n generator: %s",g);
crc(n);
printf("\n quotient: %s",q);
caltrans(n);
printf("\n transmitted frame: %s",m);
printf("\n enter recieved frame: ");
scanf("\n%s",m);
printf("CRC checking\n");
```

DATE: PAGE:

```
    crc(n);
    printf("\n last remainder : %.8", r);
    for(i=0; i<16; i++)
      if(r[i] != '0')
      flag = 1;
      else
      continue;
    if(flag == 1)
    printf("Error during transmission.");
    else
    printf("\n Recieved frame is correct");
}
void crc(int n) {
    int i, j;
    for(i=0; i<n; i++)
    temp[i] = m[i];
  for(i=0; i<16; i++)
    r[i] = m[i];
  for(i=0; i<n-16; i++) {
    if(r[0] == '1') {
        q[i] = '1';
        caltram();
    } else {
        q[i] = '0';
        shift1();
    }
  r[16] = m[17+i];
  r[17] = '\0';
  for(j=0; j<=17; j++)
    temp[j] = r[j];
}
```

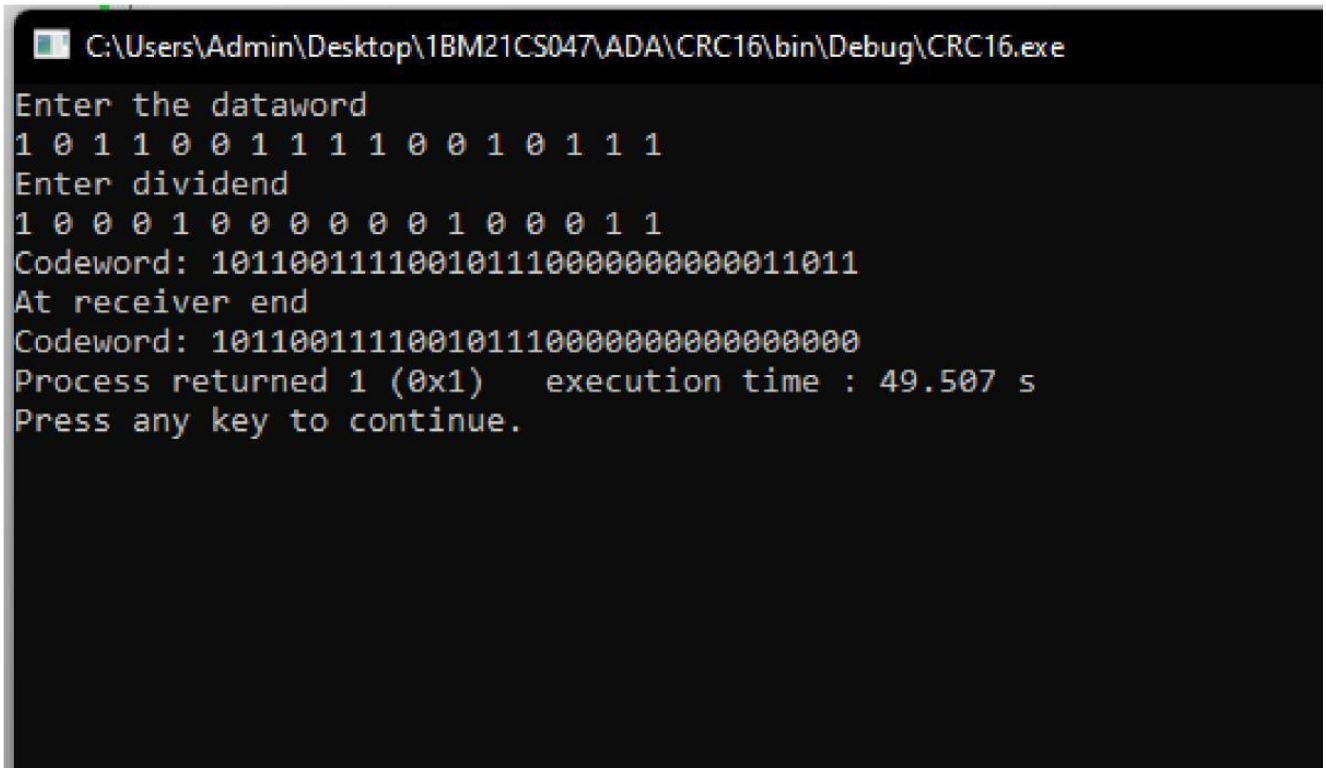
DATE: PAGE:

```
    g[n-16]='1';
}
void calram(){
    int i,j;
    for(i=1;i<=16;i++)
    r[i-1]=((int)temp[i]-48)^((int)g[i]-48)+48;
}
void shift(){
    int i;
    for(i=1;i<=16;i++)
    r[i-1]=r[i];
}
void caltrans(int n){
    int i,k=0;
    for(i=n-16;i<n;i++)
    m[i]=((int)m[i]-48)^((int)r[k++]-48)+48;
    m[i]='1'; return 0;}
```

OUTPUT-

```
enter the frame bits: 1011
message after appending 16 zeroes:
                    1011000000000000
generator: 1000100000010001
quotient: 1011
transmitted frame: 1011011000001011
crc checking
last mechanism: 0000000000000000
recieved frame is correct.
```

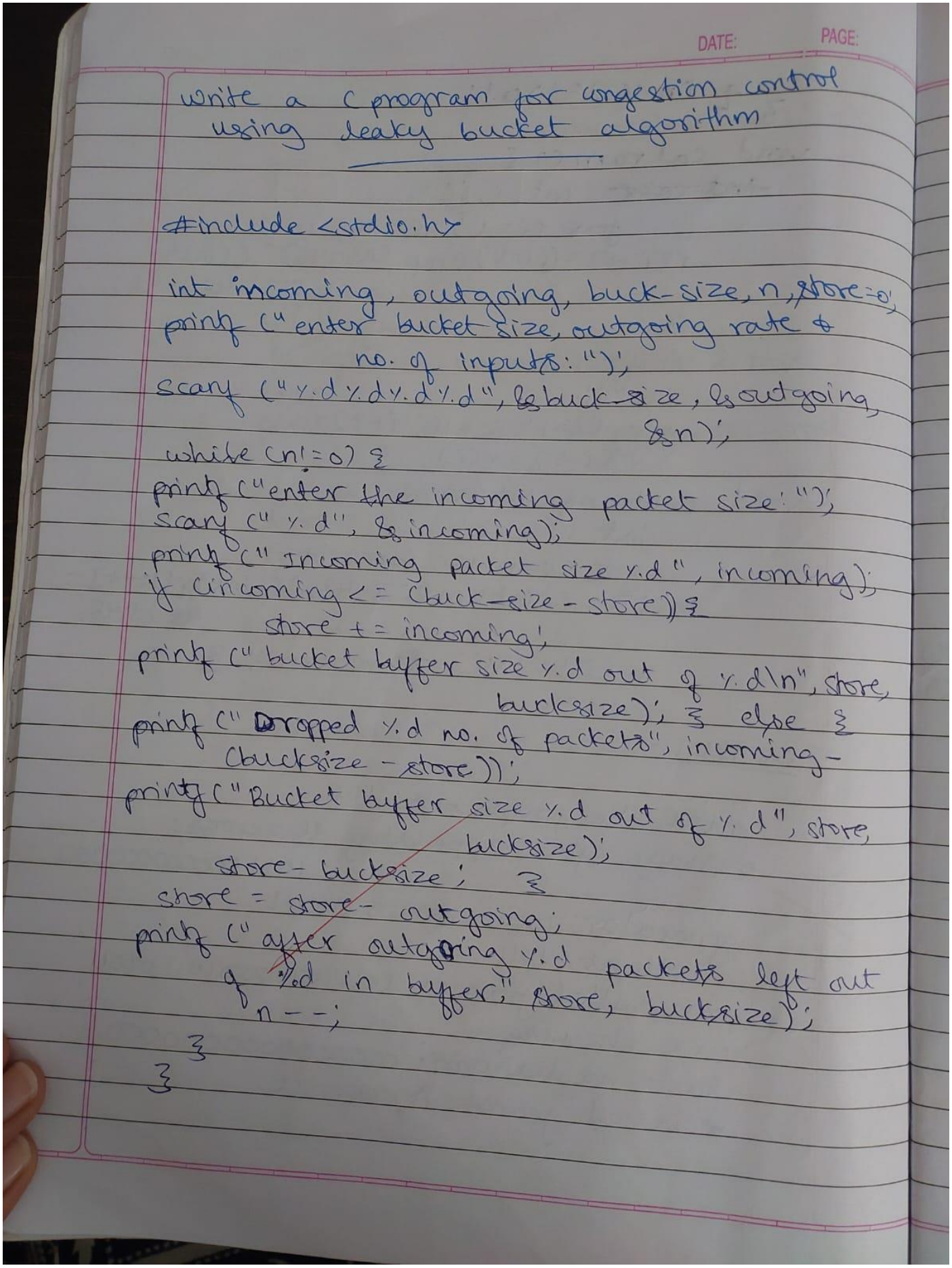
System Output-



LAB 12

Write a program for congestion control using Leaky bucket algorithm.

Observation-



DATE: PAGE:

write a C program for congestion control using leaky bucket algorithm

```
#include <stdio.h>

int incoming, outgoing, buck-size, n, store=0;
printf ("enter bucket size, outgoing rate &
        no. of inputs: ");
scanf ("%d %d %d %d", &buck-size, &outgoing,
                      &n);
while (n!=0) {
printf ("enter the incoming packet size: ");
scanf ("%d", &incoming);
printf ("Incoming packet size %d", incoming);
if (incoming <= (buck-size - store)) {
        store += incoming;
printf ("bucket buffer size %d out of %d\n", store,
                      bucksize); } else {
printf ("Dropped %d no. of packets", incoming-
        (bucksize - store));
printf ("Bucket buffer size %d out of %d", store,
                      bucksize);
    store- bucksize; }
store = store- outgoing;
printf ("after outgoing %d packets left out
    of %d in buffer", store, bucksize);
    n--;
}
}
```

DATE: PAGE:

- OUTPUT -

enter bucket size, outgoing rate & no. of inputs: 20 10 2

→ enter the incoming packet size: 30

⇒ incoming packet size: 30
dropped 10 no. of packets
buffer size 0 out of 20
after outgoing 10 packets left out of 20 in buffer

→ enter the incoming packet size: 10

⇒ incoming packet size: 10
buffer size 20 out of 20
after outgoing 10 packets out of 20 in buffer.

1/9

System Output-

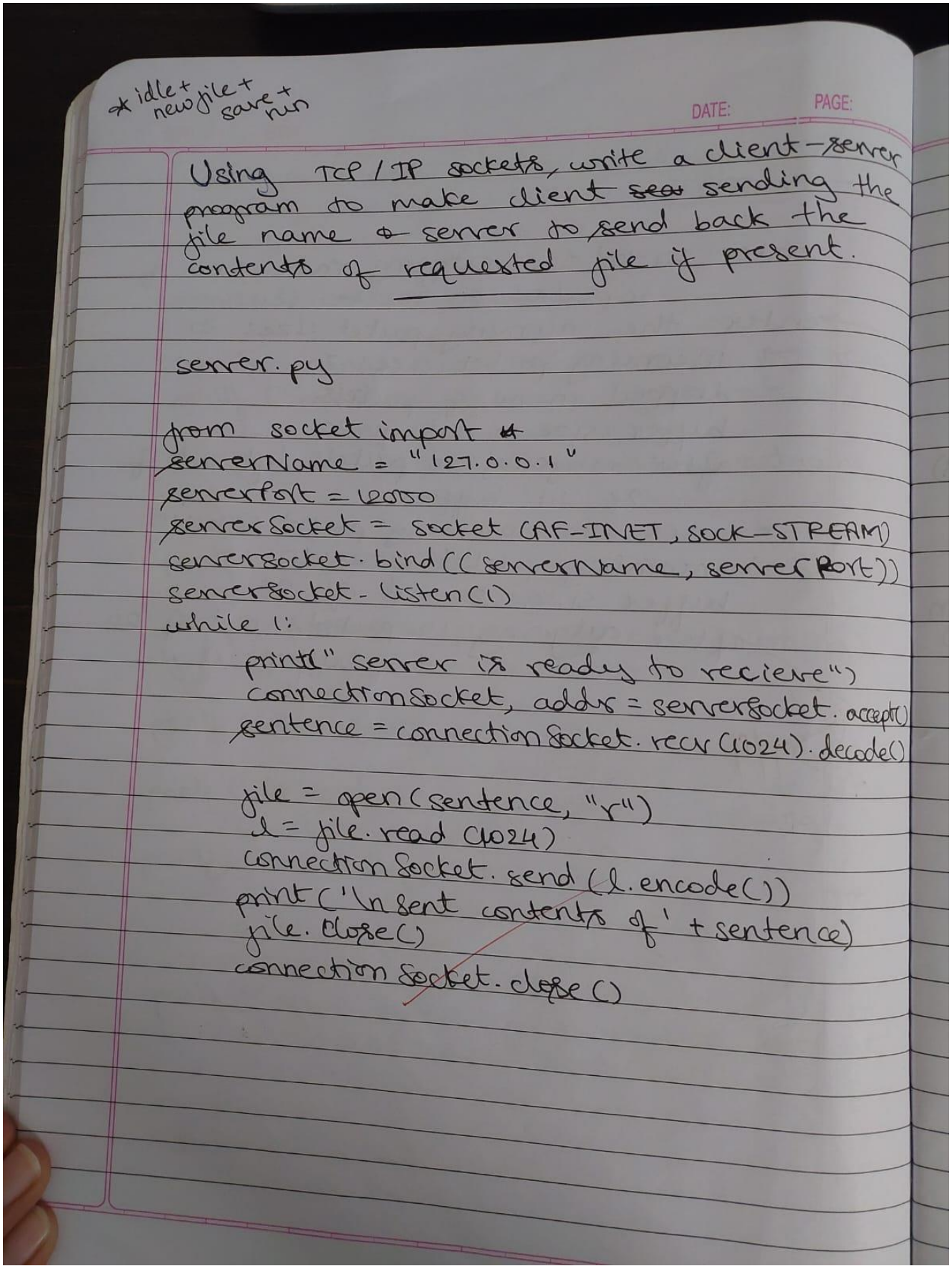
```
PS D:\VS Code> cd "d:\VS Code\OS\" ; if ($?) { gcc bucket.c -o bucket } ; if ($?) { .\bucket }
Enter Bucket size and outstream size
2000
100
Packet of 41 bytes accepted
Remaining bytes: 2000
If you want to stop input, press 0, otherwise, press 1
1
Packet of 467 bytes accepted
Remaining bytes: 1633
If you want to stop input, press 0, otherwise, press 1
1
Packet of 334 bytes accepted
Remaining bytes: 1399
If you want to stop input, press 0, otherwise, press 1
1
Packet of 500 bytes accepted
Remaining bytes: 999
If you want to stop input, press 0, otherwise, press 1
1
Packet of 169 bytes accepted
Remaining bytes: 930
If you want to stop input, press 0, otherwise, press 1
1
Packet of 724 bytes accepted
Remaining bytes: 306
If you want to stop input, press 0, otherwise, press 1
1
Packet of 478 bytes is discarded
Remaining bytes: 406
If you want to stop input, press 0, otherwise, press 1
1
Packet of 358 bytes accepted
Remaining bytes: 148
If you want to stop input, press 0, otherwise, press 1
1
Packet of 962 bytes is discarded
Remaining bytes: 248
If you want to stop input, press 0, otherwise, press 1
0
Remaining bytes: 348
Remaining bytes: 448
Remaining bytes: 548
Remaining bytes: 648
Remaining bytes: 748
```

```
Remaining bytes: 348
Remaining bytes: 448
Remaining bytes: 548
Remaining bytes: 648
Remaining bytes: 748
Remaining bytes: 848
Remaining bytes: 948
Remaining bytes: 1048
Remaining bytes: 1148
Remaining bytes: 1248
Remaining bytes: 1348
Remaining bytes: 1448
Remaining bytes: 1548
Remaining bytes: 1648
Remaining bytes: 1748
Remaining bytes: 1848
Remaining bytes: 1948
Remaining bytes: 2000
PS D:\VS Code\OS>
```

LAB 13

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

Observation-



* idle + new file + save + run

DATE: PAGE:

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

server.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while 1:
    print(" server is ready to 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()
```

DATE: PAGE:

client.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:\n")
print (fileContents)
clientSocket.close()
```

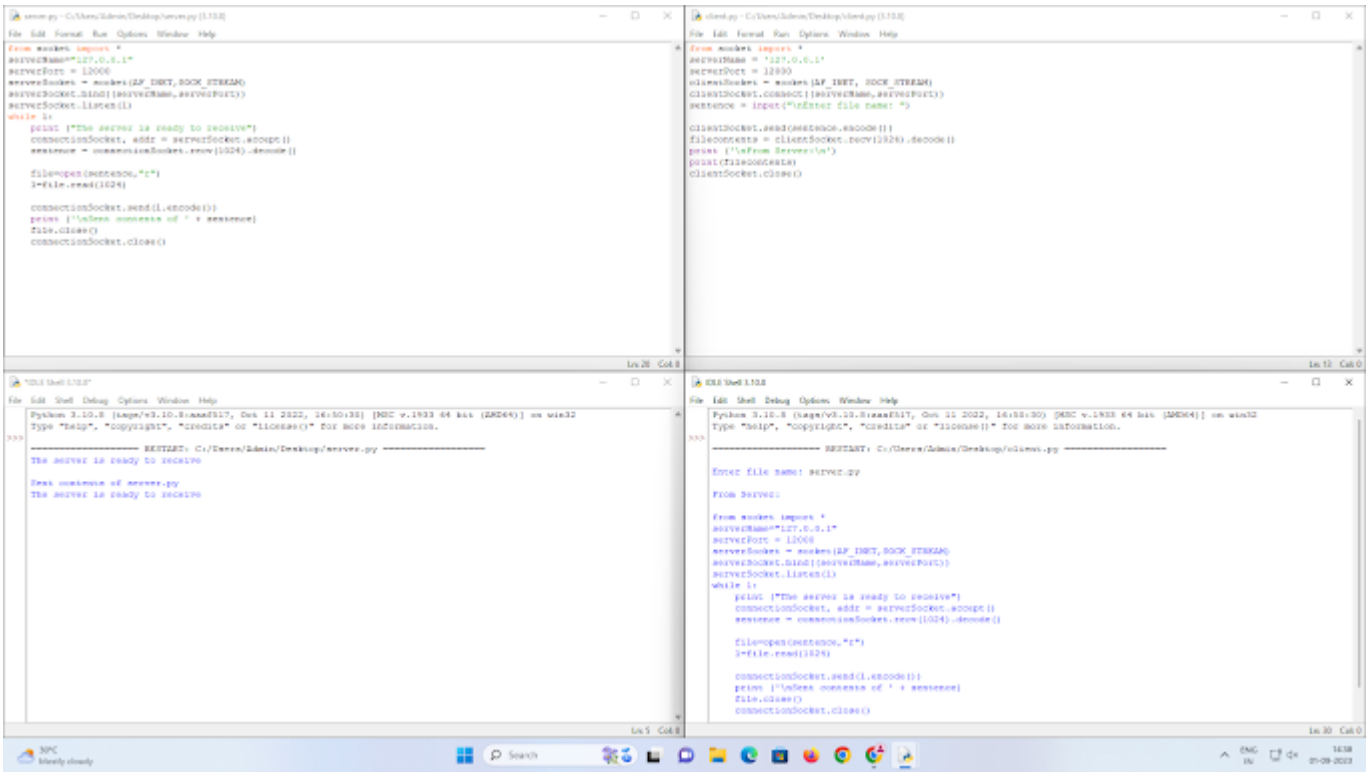
-OUTPUT-

1) server is ready to recieve
sent contents to server.py
server is ready to recieve.

2) enter file name: server.py
~~from server:~~
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_IN

(contents of server.py, o/p).

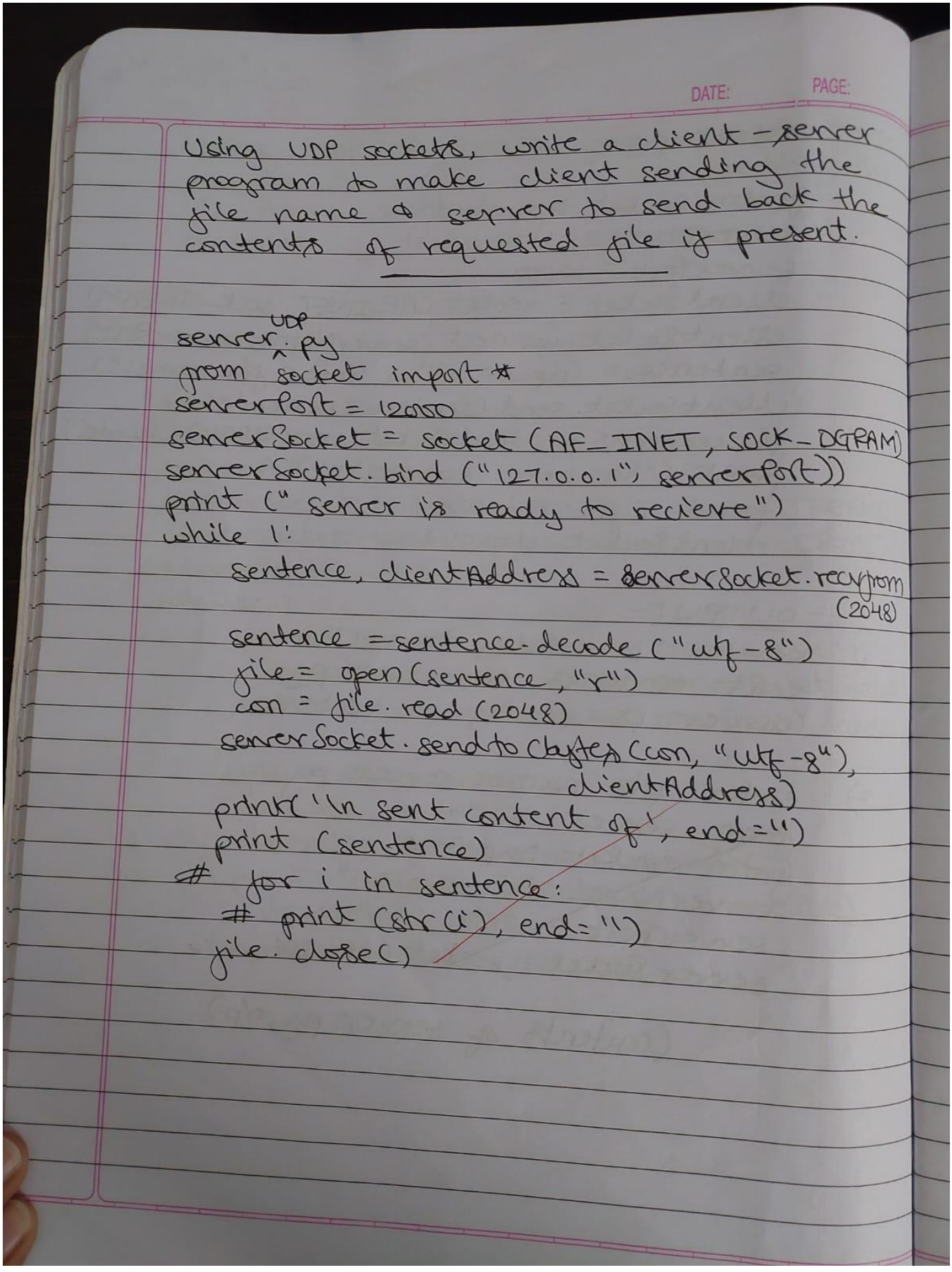
System Output-



LAB 14

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

Observation-



DATE: PAGE:

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

server.py (UDP)

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print("server is ready to recieve")
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('\n sent content of', end=' ')
    print(sentence)
#   for i in sentence:
#       print(str(i), end='')
    file.close()
```

DATE: PAGE:

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 Reply from Server:\n')
print(fileContents.decode("utf-8"))
# for i in fileContents:
    # print(str(i), end="")
clientSocket.close()
clientSocket.close()
```

- OUTPUT -

1) Server is ready to recieve
   Sent contents of serverUDP.py

2) enter file name: serverUDP.py
   (contents of serverUDP.py O/P)

System Output-