VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

RIA JAIN (1BM21CS163)

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
JUN-2023 to SEP-2023
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 **RIA JAIN(1BM21CS163) ,** who is a 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.

Dr. Nandini Vineeth
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

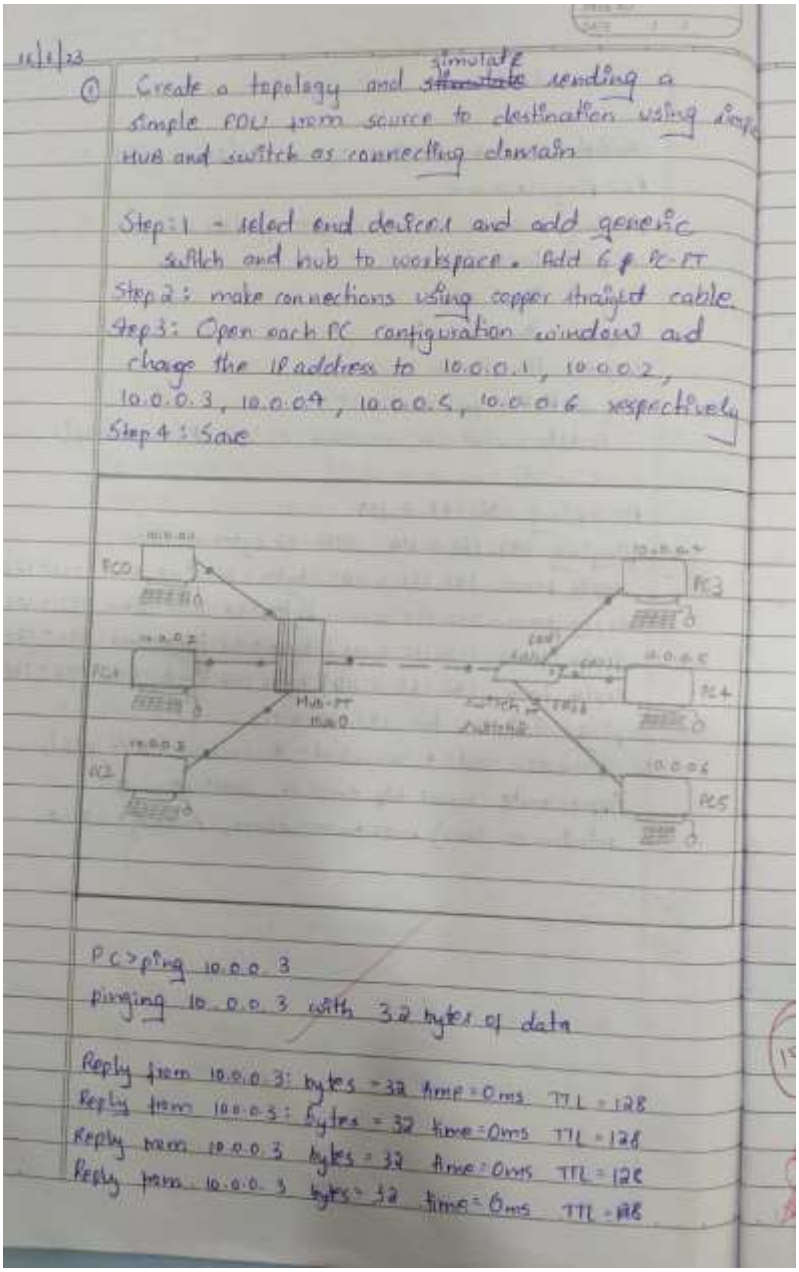
`

Index

WEEK 1

Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping messages.

OBSERVATION:



11/1/23

① Create a topology and simulate sending a simple PDU from source to destination using simple hub and switch as connecting domain

Step:1 - select and design and add generic switch and hub to workspace. Add 6 PC-PT

Step 2: make connections using copper straight cable

Step 3: Open each PC configuration window and change the IP address to 10.0.0.1, 10.0.0.2, 10.0.0.3, 10.0.0.4, 10.0.0.5, 10.0.0.6 respectively

Step 4: Save

PC>ping 10.0.0.3

pinging 10.0.0.3 with 32 bytes of data

Reply from 10.0.0.3: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.3: bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 128

Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 128

Ping statistics for 10.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% Loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms

Scenario : 1

① Sending ~~packet~~ ~~package~~ packet from PC0 to PC1 in hub.
The package is sent from PC0 to the HUB, The HUB sends the package to PC1 and PC2. Since PC1 is the receiver it receives the ~~package~~ packet and sends the acknowledgement back to ~~PC~~ HUB which sends it back to PC0 that is the sender and other PC's.

Scenario : 2

① Sending ~~package~~ packet from PC3 to PC4 in switch.
The package is sent from PC3 to switch PC4 which receives the packet and since it is the receiver, it sends the acknowledgment back to switch that sends to PC3.

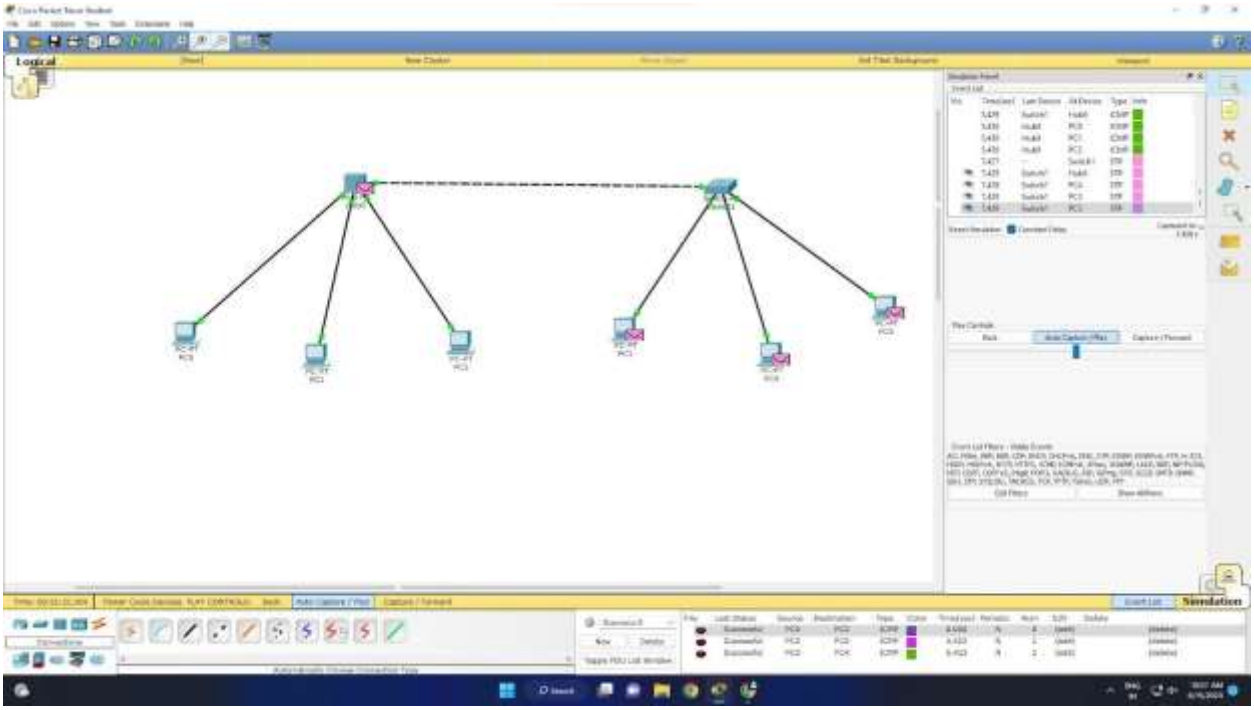
Scenario : 3

① Sending b/w PC's connected to switch and HUB. Packet is sent from PC0 to PC4. The HUB sends the packet to PC1, PC2 and the switch. The switch sends the packet to PC3 and then PC4 which is the receiver sends the acknowledgment back to switch which sends it back to the HUB which forwards it all the PC's connected to the HUB i.e PC0, PC1 and PC2.

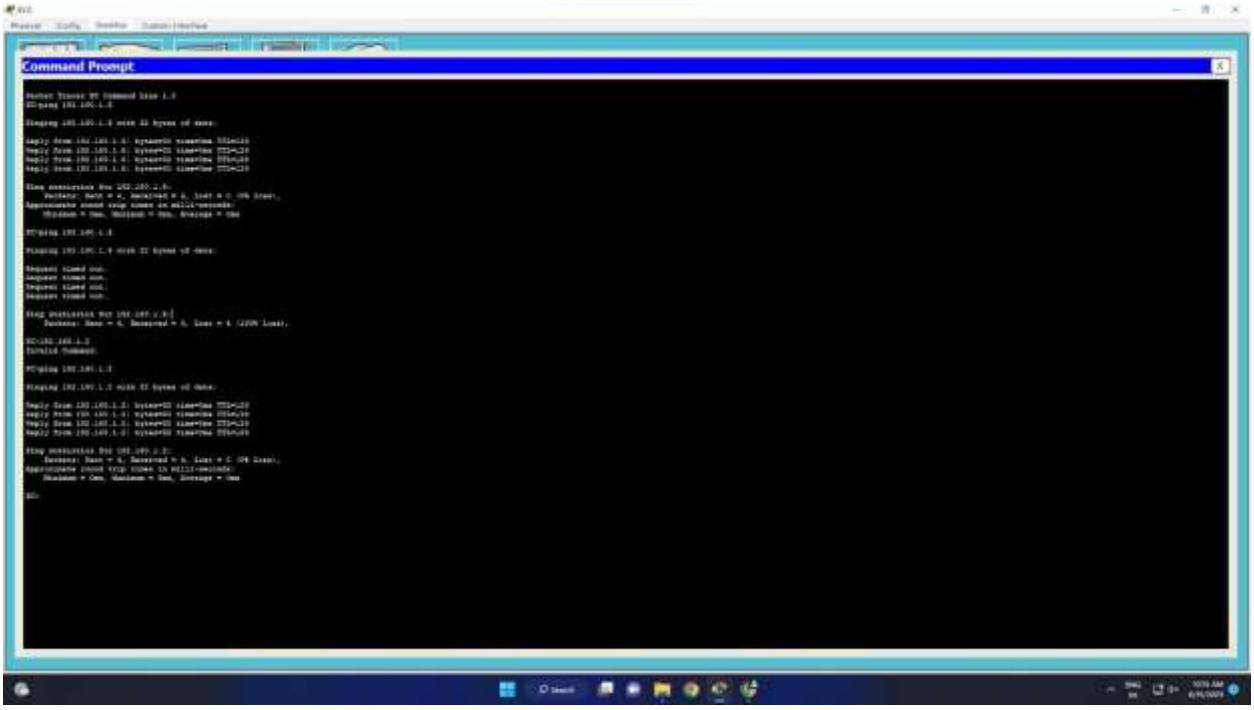
10/10

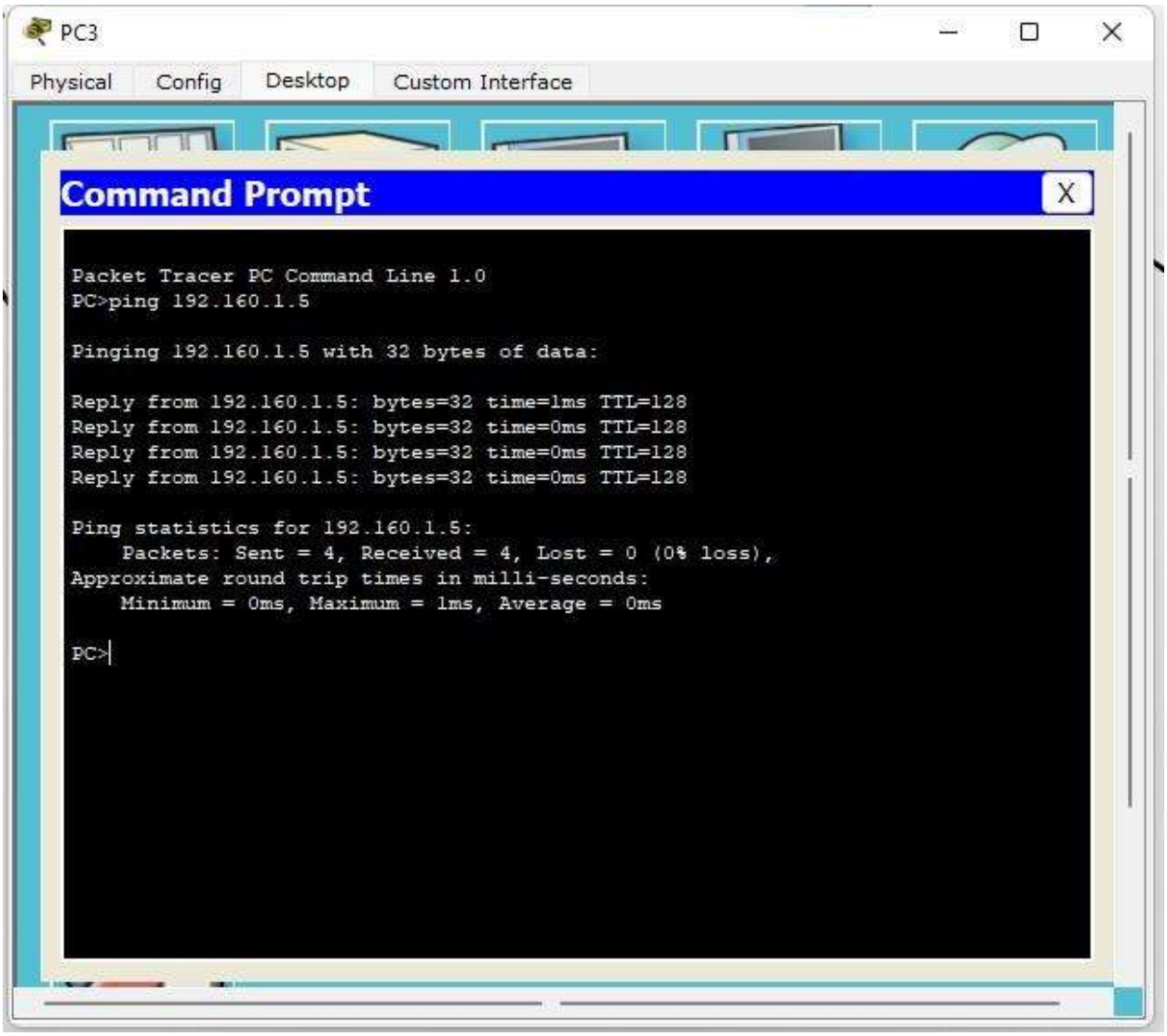
16/16

TOPOLOGY:



OUTPUT:




PC3
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 192.160.1.5
Pinging 192.160.1.5 with 32 bytes of data:
Reply from 192.160.1.5: bytes=32 time=1ms TTL=128
Reply from 192.160.1.5: bytes=32 time=0ms TTL=128
Reply from 192.160.1.5: bytes=32 time=0ms TTL=128
Reply from 192.160.1.5: bytes=32 time=0ms TTL=128
Ping statistics for 192.160.1.5:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
PC>

WEEK 2

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

OBSERVATION:

23/6/23

Q) Configure default route, static route to router.

Sol: Steps:

1. Add 2 PCs & connect them to generic router
2. Configure the PCs by setting their IP addresses to 10.0.0.1 & 20.0.0.1 respectively
3. set the gateways as 10.0.0.2 & 20.0.0.2 respectively
4. Go to cmd line interface in router and enter 'no' for continue with configuration dialogue choice
5. Type command 'config terminal' and enter. Continue by typing the command 'interface fastEthernet 0/0'. follow it by entering command 'ip address 10.0.0.2 255.0.0.0'. Type command 'no shut' and establish the connection.
6. Repeat same for PC 20.0.0.1.
7. Repeat same for PC3 and PC4
8. Connect router 1 and router 2 to router 3.
9. Routers to be connected through serial DTE and PC to router should be connected through copper cross over.
10. Router 1 CLI type -
    - → enable
    - → config t
    - → interface serial 2/0
    - → ip address 50.0.0.2 255.0.0.0
    - → no shut

    III^ly for 50.0.0.1 255.0.0.0

    The connection b/w router 1 & 3 is now active
11. Repeat step 10 for router 2 and 3
12. Go to router 1 CLI & type show ip route.
13. we now connect routers to each other by typing

    ip route 30.0.0.0 255.0.0.0 50.0.0.1

    ip route 40.0.0.0 255.0.0.0 40.0.0.1 for router 1

PAGE NO.
DATE / /

ip route 10.0.0.0 255.0.0.0 60.0.0.1
ip route 20.0.0.0 255.0.0.0 60.0.0.1 for router 2

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 for rater 3

(14) PC> ping 30.0.0.1
pinging 30.0.0.1 with 32 bytes of data:

Reply from 20.0.0.2 : destination host unreachable
" " " " " "
" " " " " "
" " " " " "

ping statistics for 30.0.0.1:
packects : sent = 4, Received = 0, lost = 4 (100% loss)

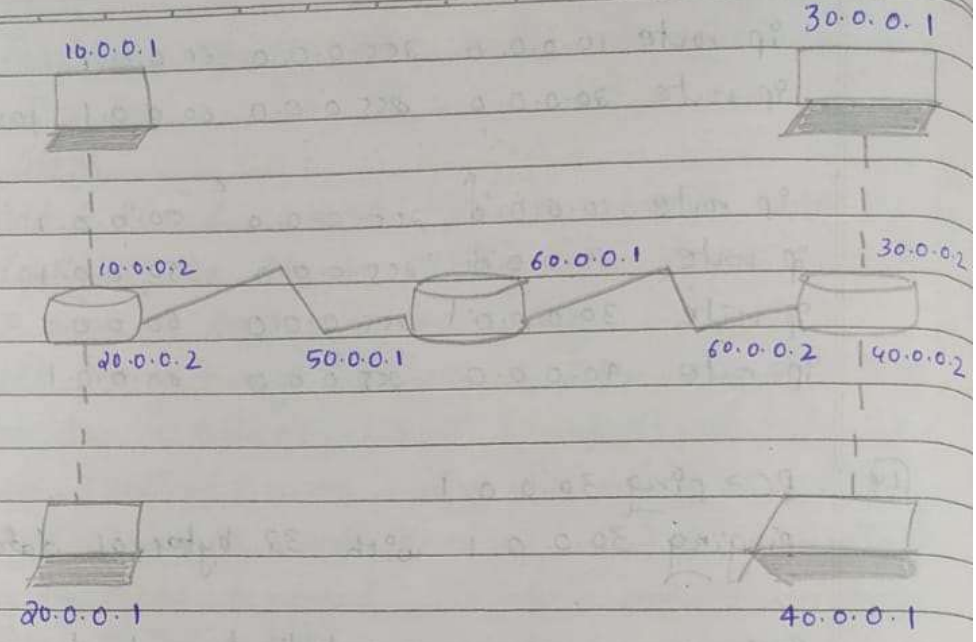
This is before statically connecting PCs.

(15) 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=2 ms TTL=125
" " " " " "
" " " " " "
" " " " " "

show IP route?

Topology →

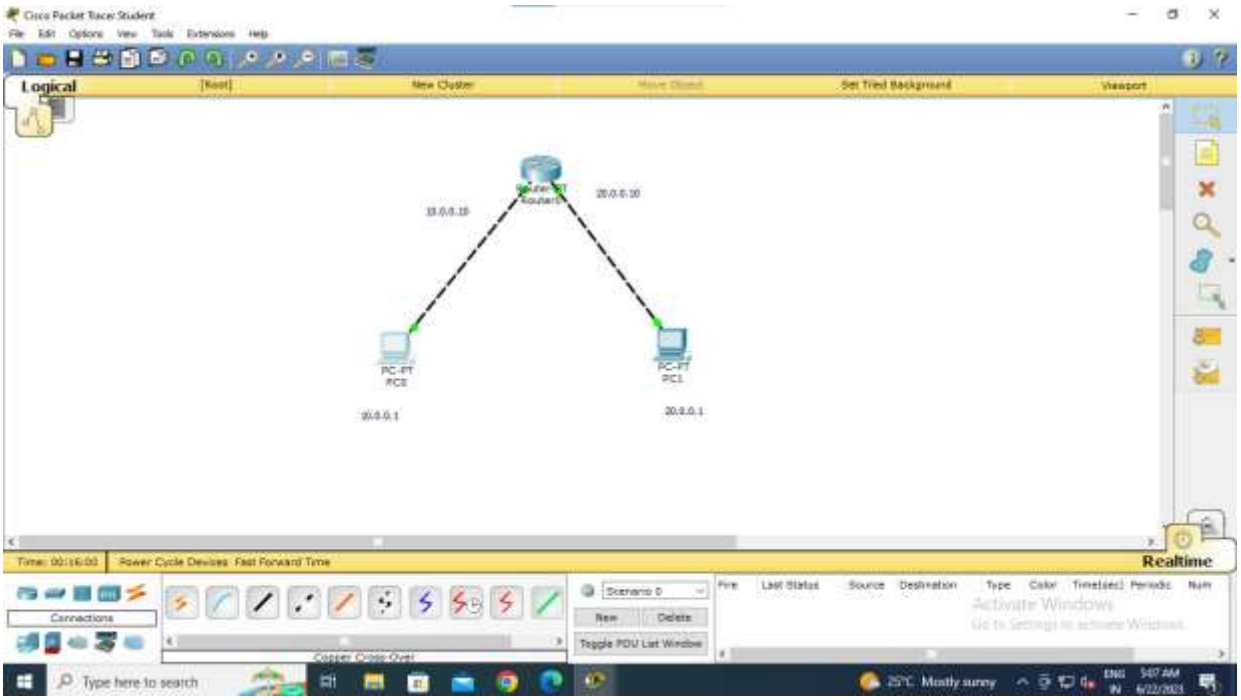


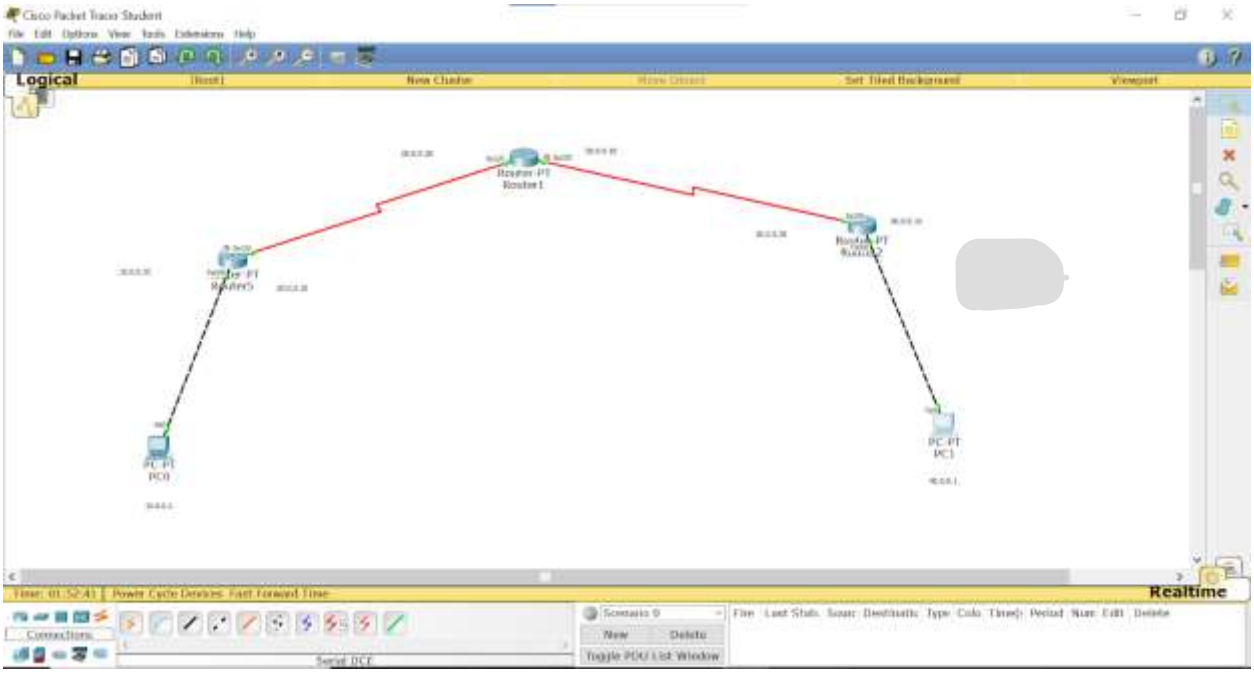
★ Commands -

* no → enter (twice)
enable → enter
config terminal → enter
interface fastEthernet 0/0 → enter
ip address 10.0.0.2 255.0.0.0 → enter
no shut down → enter.

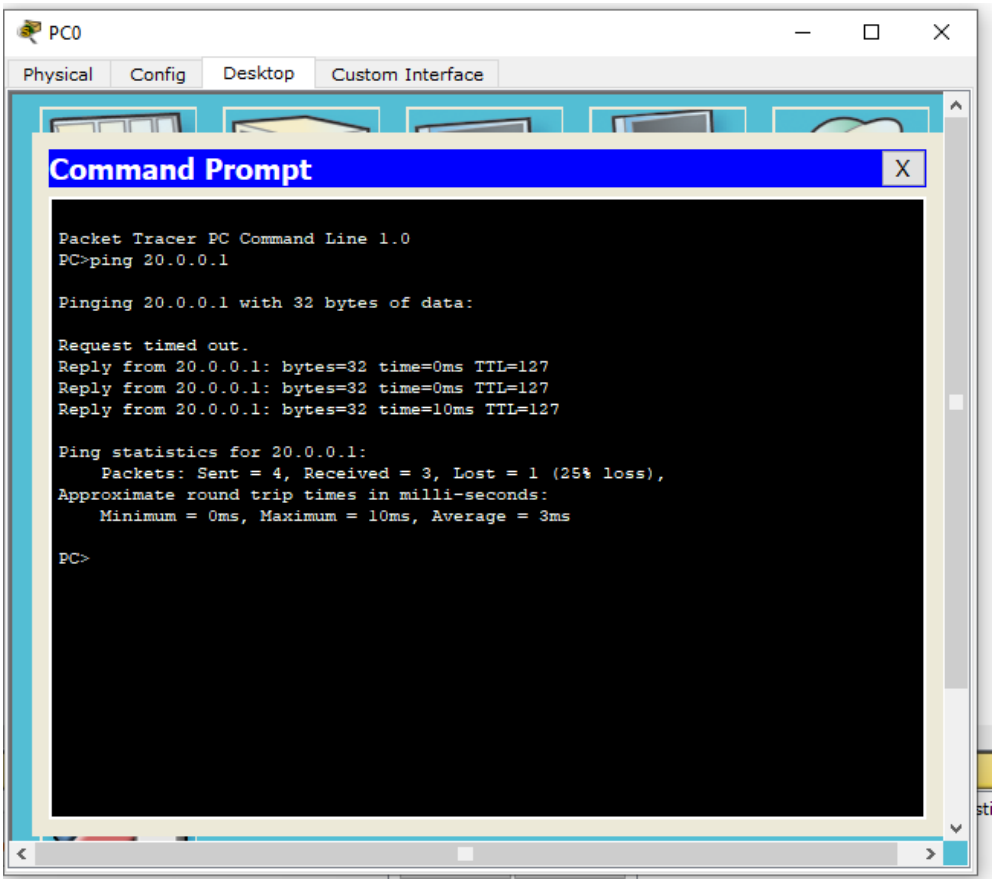
for router connection -
* same
interface serial 2/0 → enter.

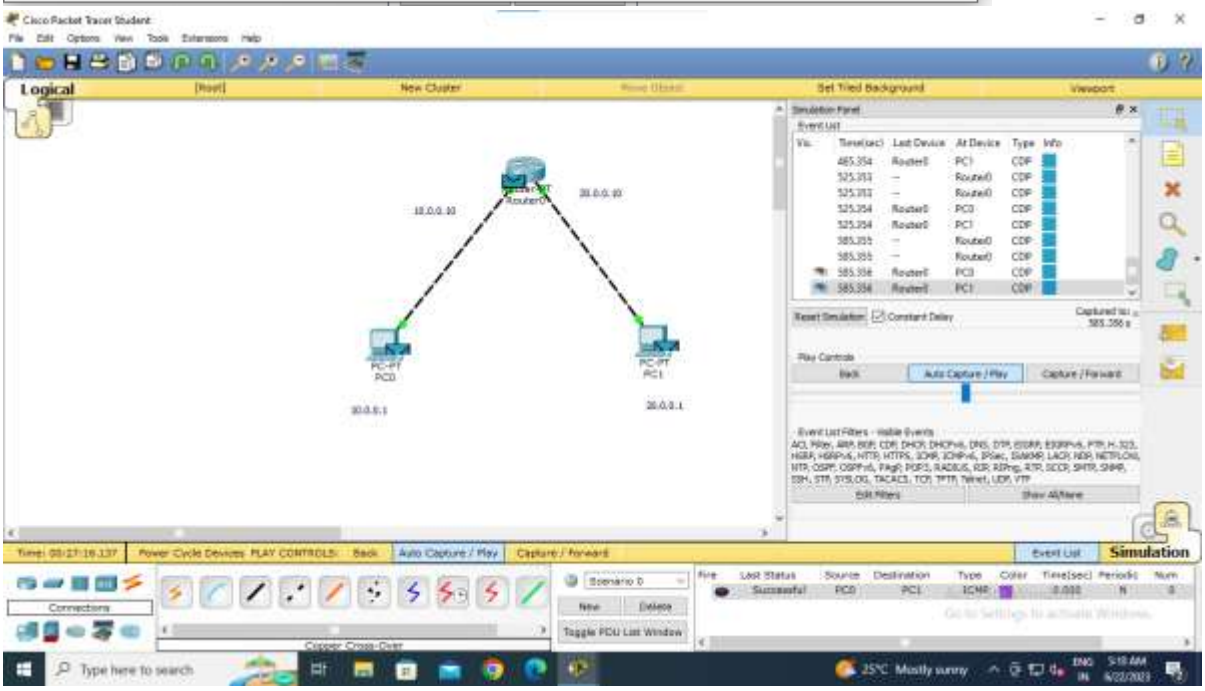
TOPOLOGY:


Cisco Packet Tracer Student
Logical
Realtime
PC-PT
PC0
PC-PT
PC1
10.0.0.1
20.0.0.1


Cisco Packet Tracer Student
Logical
Realtime
Router-PT
Router1
PC-PT
PC0
PC-PT
PC1

OUTPUT:


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:
Request timed out.
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=0ms TTL=127
Reply from 20.0.0.1: bytes=32 time=10ms TTL=127
Ping statistics for 20.0.0.1:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 10ms, Average = 3ms
PC>


Logical
Simulation

PC0

Physical | Config | Desktop | Custom Interface

Command Prompt

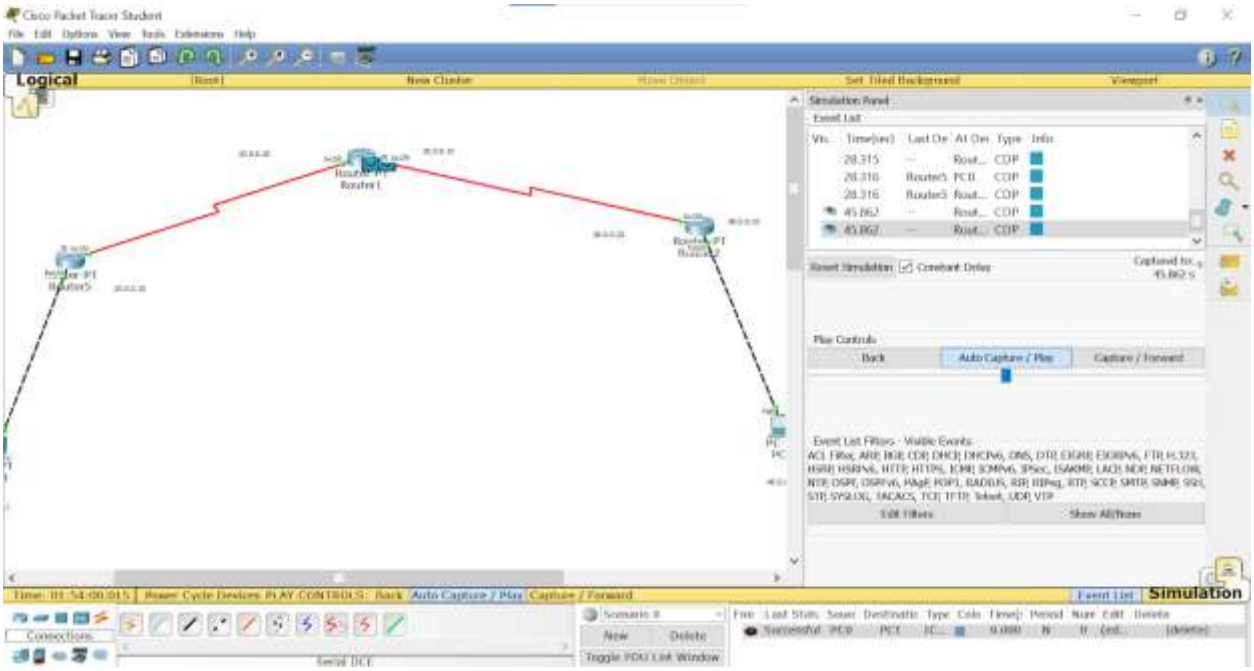
```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Request timed out.

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100%
loss),

PC>
```

PC1

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=2ms TTL=125
Reply from 10.0.0.1: bytes=32 time=8ms TTL=125
Reply from 10.0.0.1: bytes=32 time=2ms TTL=125
Reply from 10.0.0.1: bytes=32 time=2ms TTL=125

Ping statistics for 10.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 8ms, Average = 3ms

PC>
```


Cisco Packet Tracer Student
Logical
Simulation Panel
Event List
Play Controls
Back
Auto Capture / Play
Capture / Forward
Simulation

WEEK 3

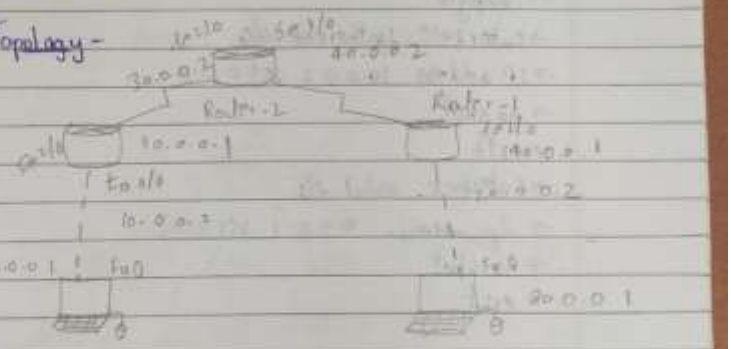
Configure default route, static route to the Router.

OBSERVATION:

30/6/23

8) Configure default route, static route to the routers.

Topology -



★ Procedure -

① Drag and drop 2 PC's and 3 routers from end devices. Connect the 1 router for each of the PC's and connect the other two router as shown in the topology.

② Set the IP address of PC0 as 10.0.0.1 and IP address of PC1 as 20.0.0.1, set the gateway as 10.0.0.2 and 20.0.0.2 for two PC's respectively.

③ now configure the IP address of the ports in router 0 and router 1 using following commands:

DATE / /

→ enable
→ config t
→ interface fastthernet 0/0
→ IP address 10.0.0.2 255.0.0.0
→ no shut
→ exit
→ interface serial 2/0
→ ip address 30.0.0.1 255.0.0.0
→ no shut
→ exit.

→ As router 0 and 1 are connected to only one side we perform default routing using following commands:
→ enable
→ config t
→ ip route 0.0.0.0 0.0.0.0 30.0.0.2

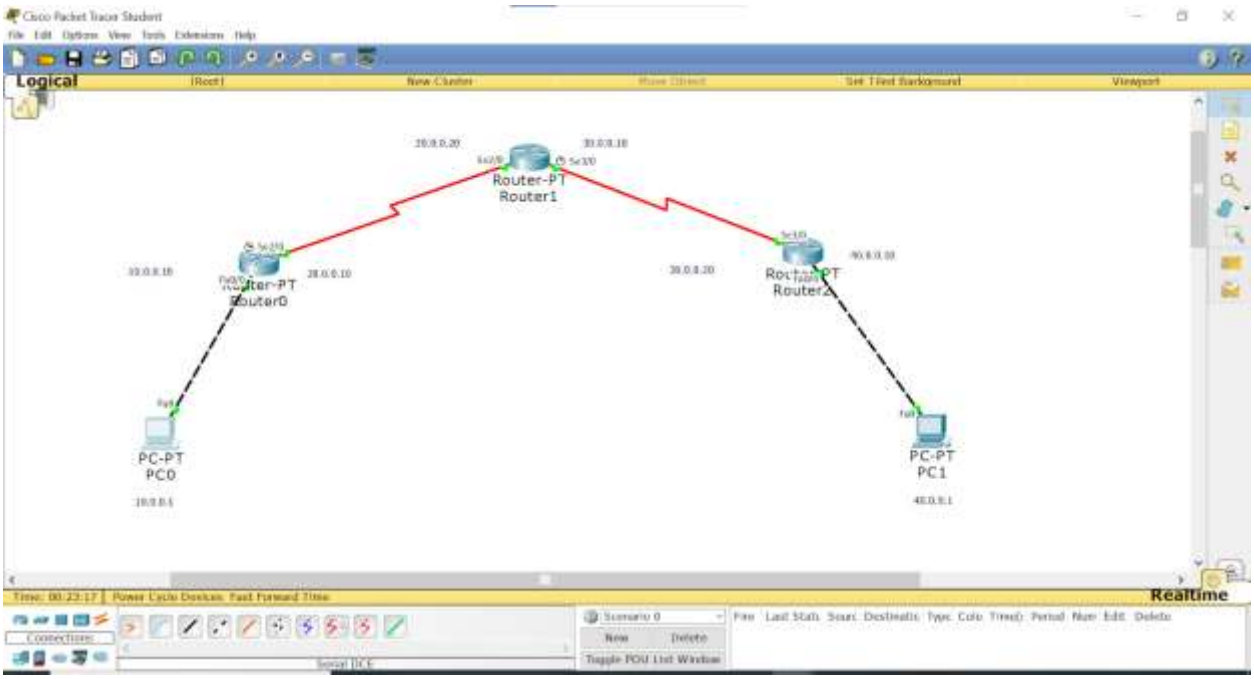
for router 1:
> ip route 0.0.0.0 0.0.0.0 40.0.0.2
for router 2:
→ ip route 10.0.0.0 255.0.0.0 50.0.0.2
→ ip route 20.0.0.0 255.0.0.0 40.0.0.2
→ exit.

10/10

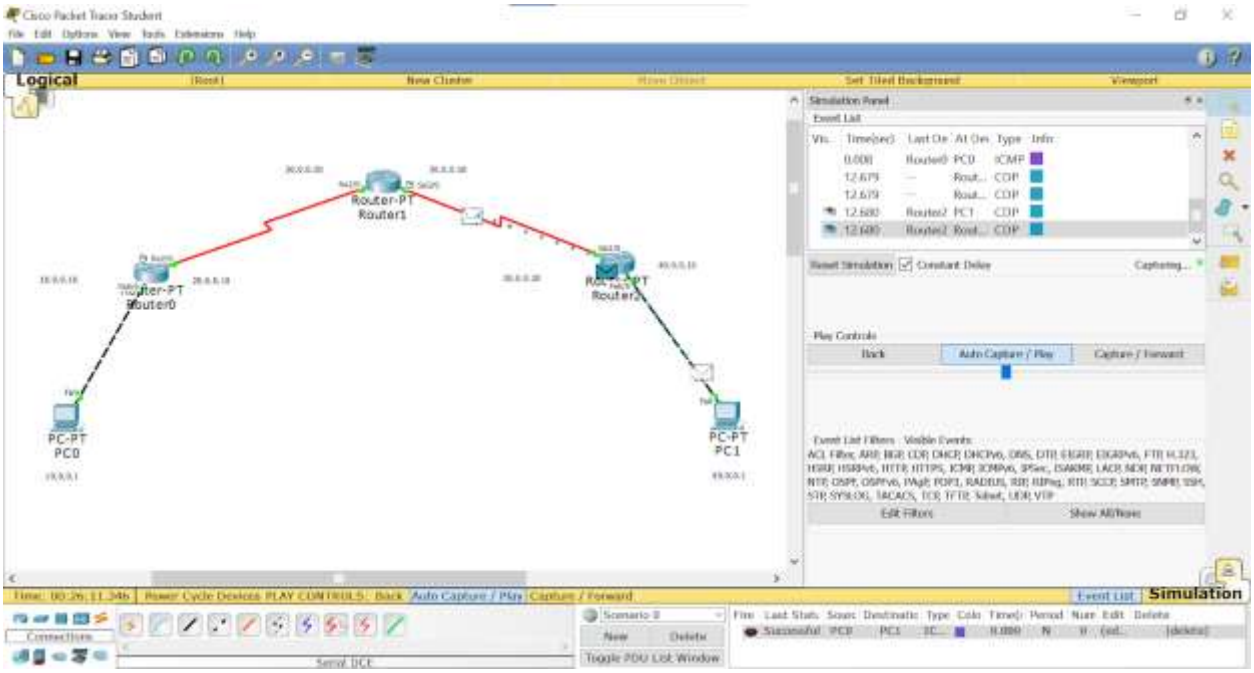
Ping commands:
PC > ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data
Reply from 20.0.0.1 bytes = 32 time = 4 ms
TTL = 125
packets sent = 4, received = 4, lost = 0 (0% loss)
minimum = 4 ms, maximum = 25 ms, Avg = 16 ms

17/7/23

TOPOLOGY:



OUTPUT:



PC0

Physical | Config | Desktop | Custom Interface

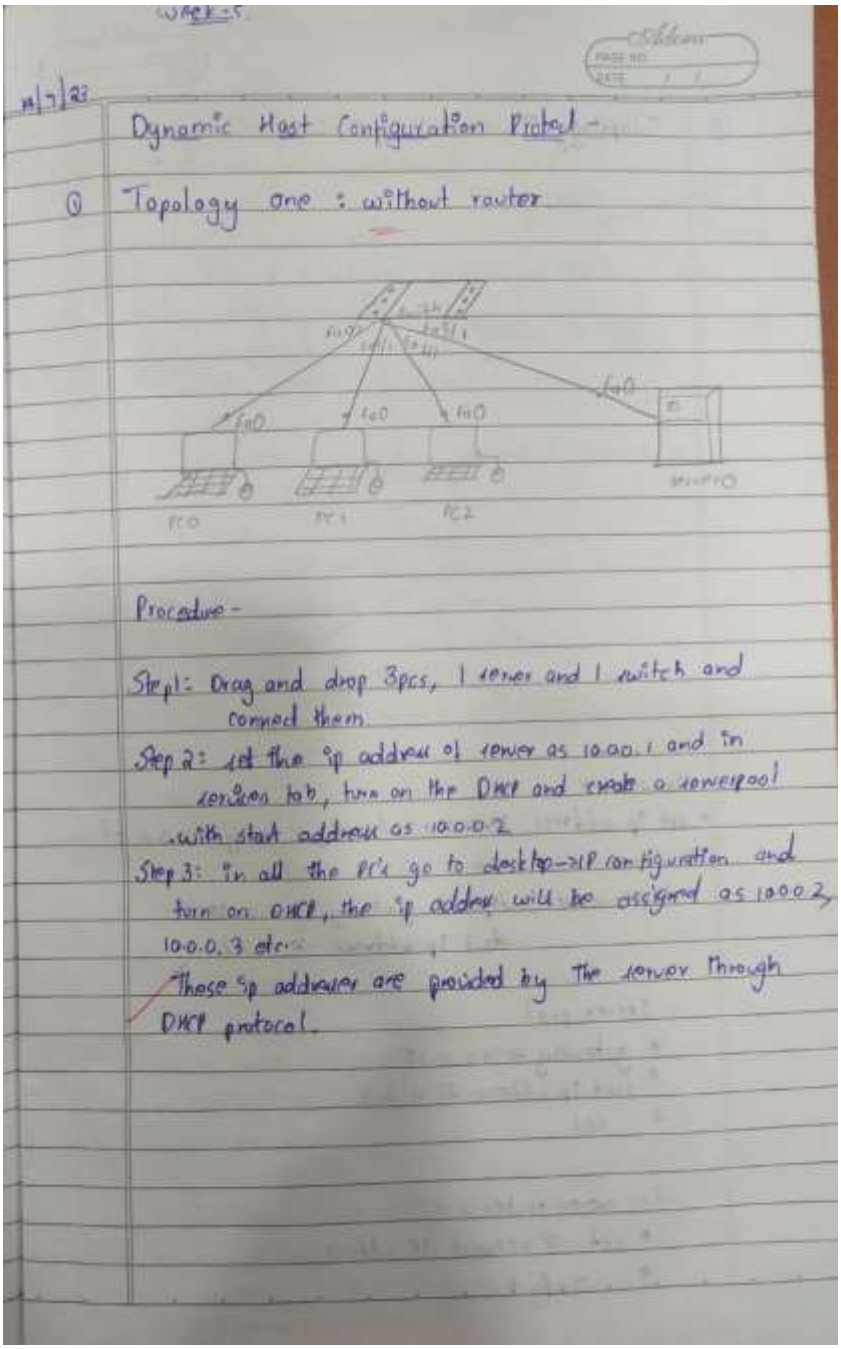
Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>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=16ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 16ms, Average = 6ms

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=21ms TTL=125
Reply from 40.0.0.1: bytes=32 time=9ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=4ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 21ms, Average = 9ms

PC>
```

WEEK 4

Configure DHCP within a LAN and outside LAN.

OBSERVATION:



WEEK-5

11/7/23

Dynamic Host Configuration Protocol -

① Topology one : without router

Procedure -

Step1: Drag and drop 3pcs, 1 server and 1 switch and connect them.

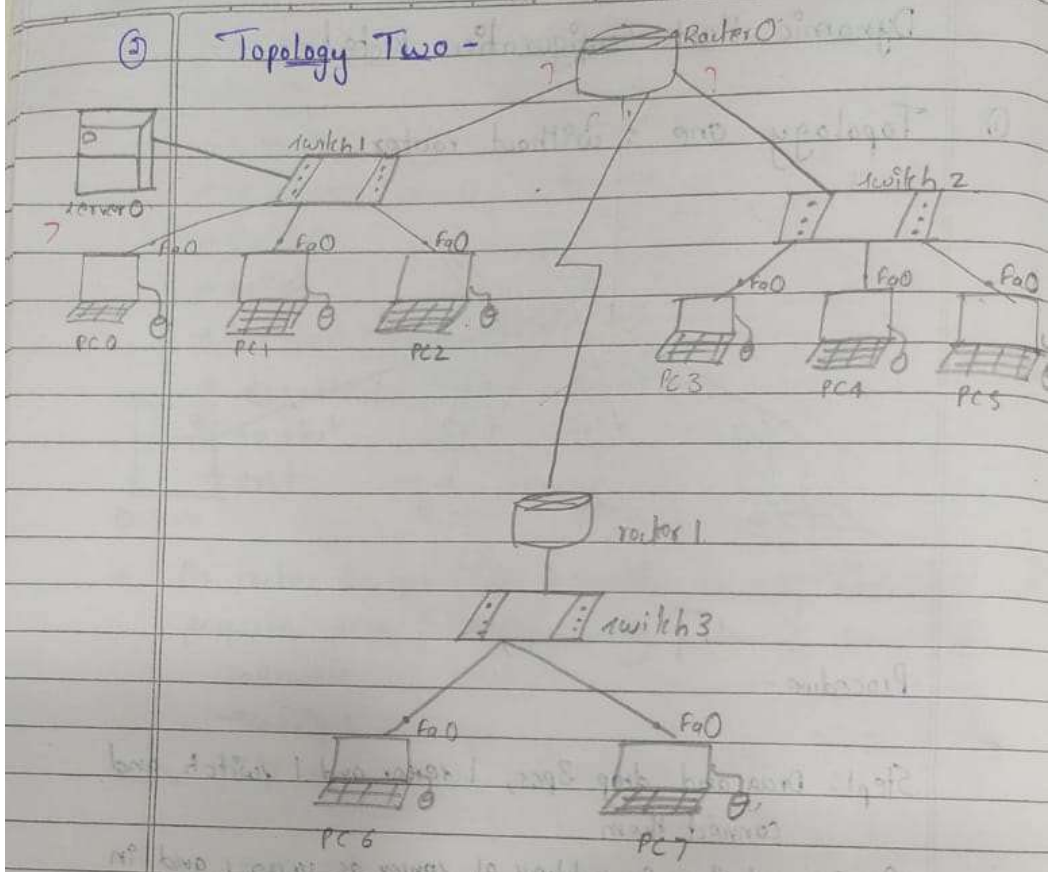
Step 2: set the ip address of server as 10.0.0.1 and in services tab, turn on the DHCP and create a serverpool with start address as 10.0.0.2

Step 3: in all the PC's go to desktop->IP configuration and turn on DHCP, the ip address will be assigned as 10.0.0.2, 10.0.0.3 etc.

Those ip addresses are provided by the server through DHCP protocol.

PAGE NO.
DATE / /

② Topology Two -



In server

* set ip address 10.0.0.1 (config → setting → 10.0.0.25)
* server → DHCP → on
* server → DHCP → default gateway 10.0.0.25
  start ip address 10.0.0.2

Server pod 1 ~~20.0.0.25~~

* gateway → 10.0.0.25
* Start ip address 20.0.0.2
* add

In ~~server~~ router 0 -

* set 2 network IP address
* config t.

PAGE NO.
DATE / /

* interface fastethernet 4/0
* ip address 10.0.0.25 255.0.0.0
* no shut.
(III'y 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 30.0.0.20

In router 1 -

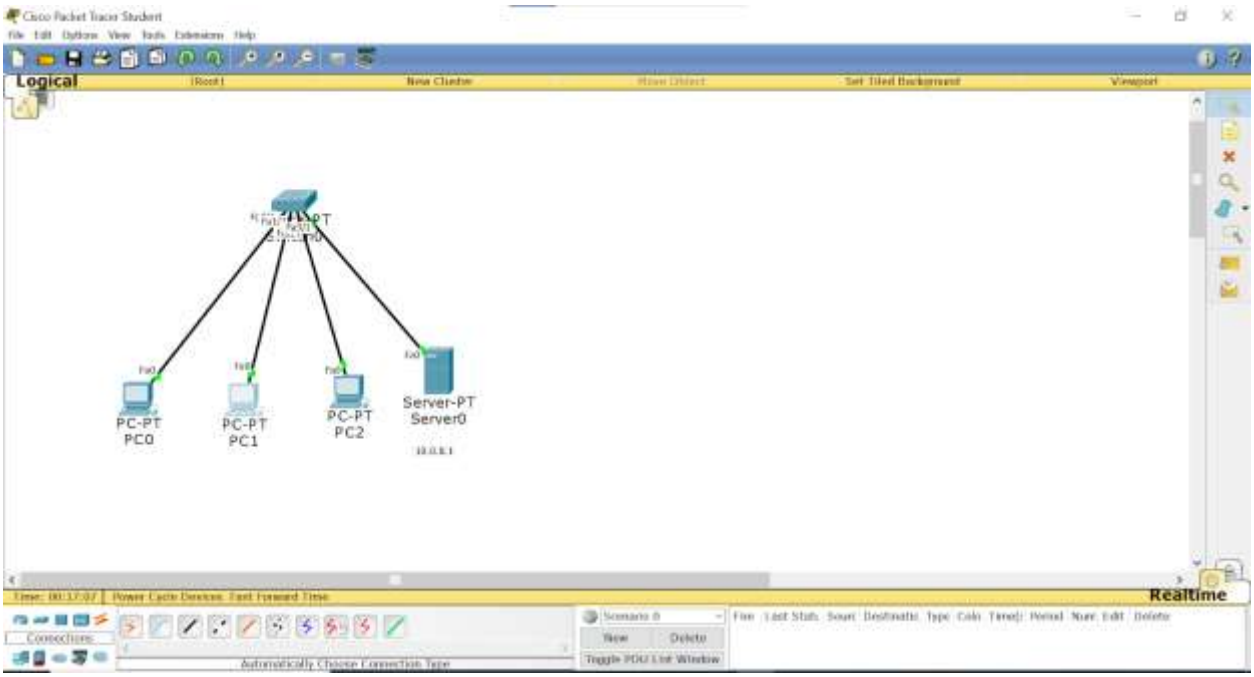
* set up address
config t
interface fastethernet 0/0
ip address 40.0.0.26 255.0.0.0
no shut
* config t
interface serial 2/0
ip address 30.0.0.26 255.0.0.0
no shut

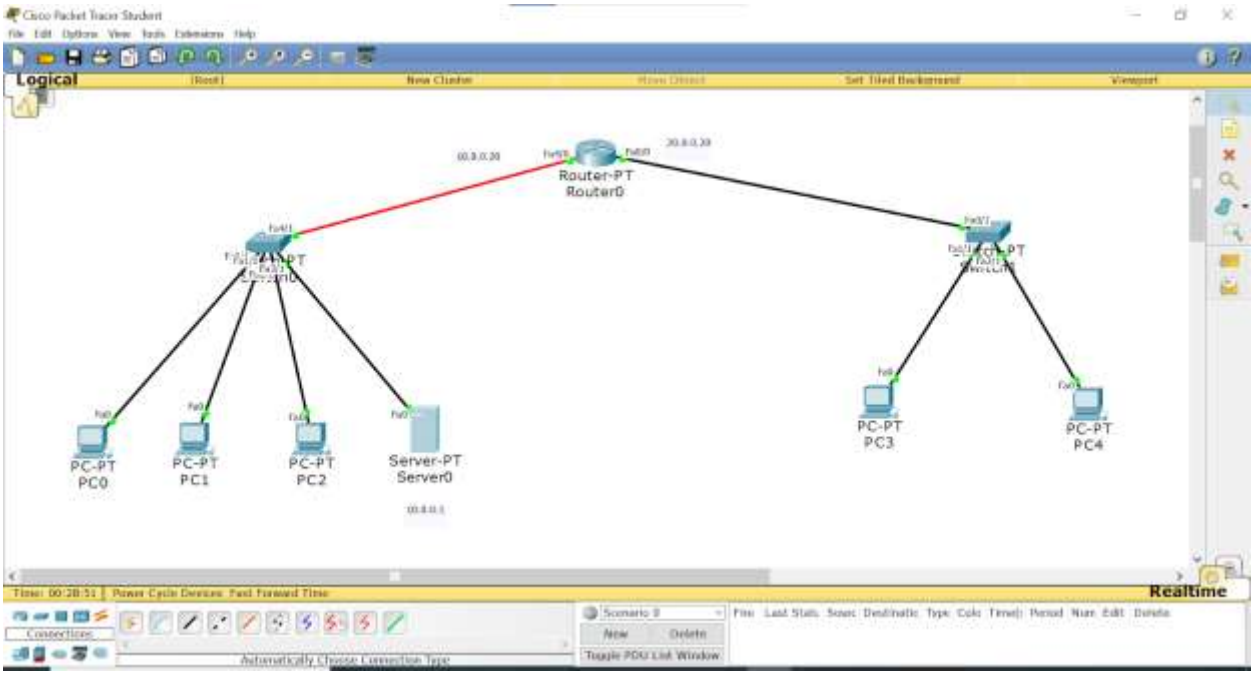
→ Static routing for 10 & 20 network
config t
ip route 10.0.0.0 255.0.0.0 30.0.0.25
ip route 20.0.0.0 255.0.0.0 30.0.0.25
no shut.

9/10
17/7/23

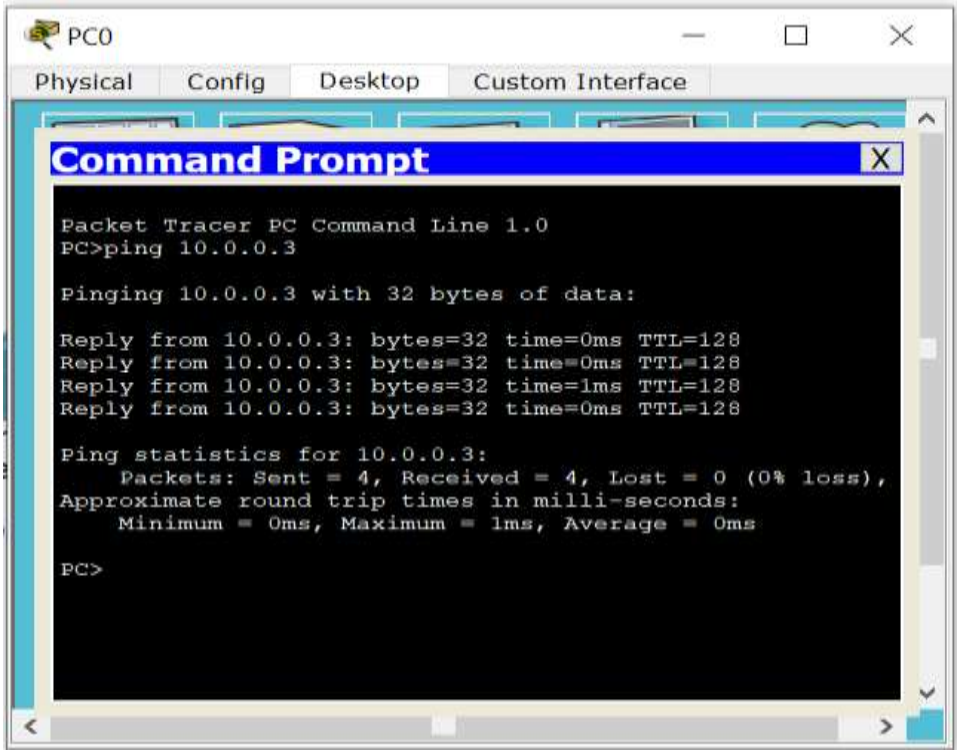
→ setting helper address
config t
interface fastethernet 0/0
ip-helper address (0.0.0.0)
no shut

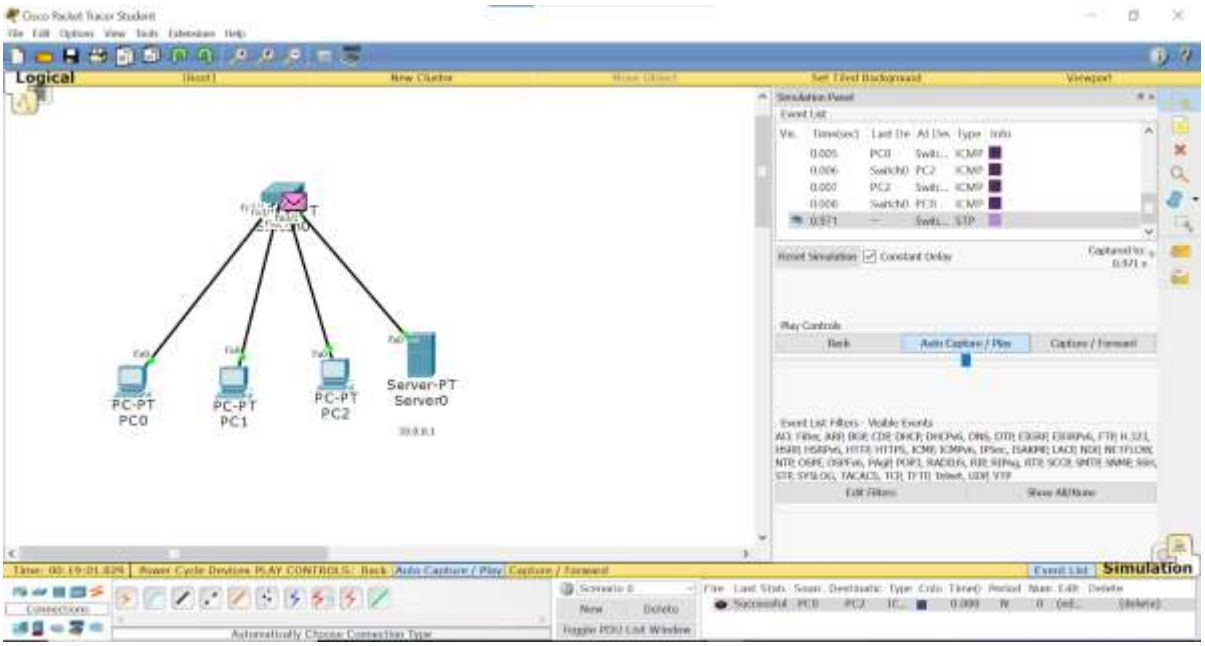
TOPOLOGY:





OUTPUT:


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128
Reply from 10.0.0.3: bytes=32 time=1ms TTL=128
Reply from 10.0.0.3: bytes=32 time=0ms TTL=128
Ping statistics for 10.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 1ms, Average = 0ms
PC>

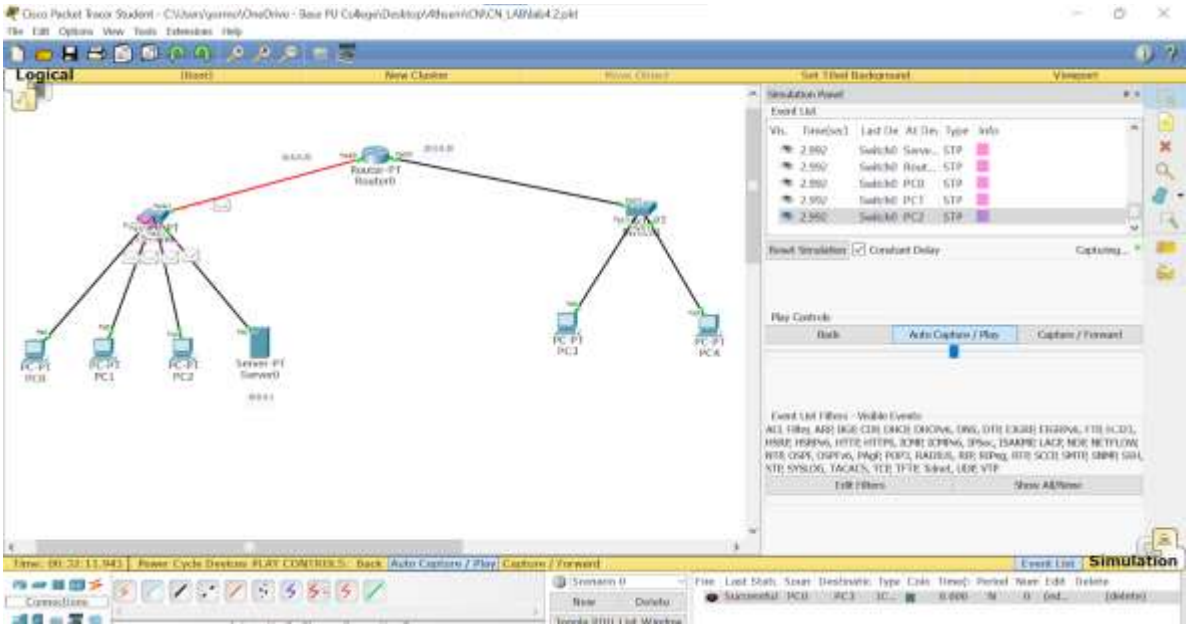

Logical
PC-PT
PC0
PC-PT
PC1
PC-PT
PC2
Server-PT
Server0
Simulation

PC0

Physical | Config | Desktop | Custom Interface

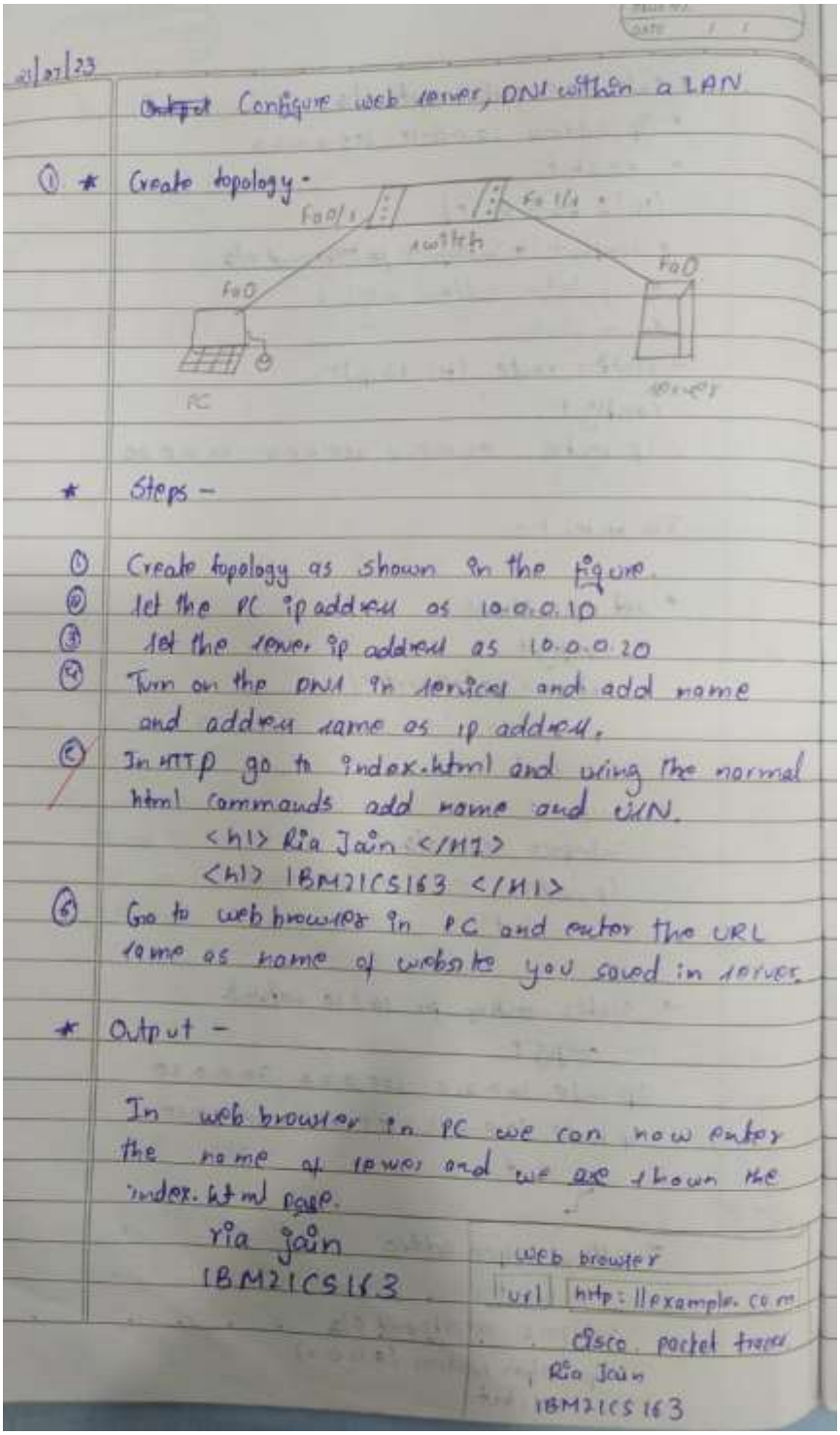
Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 20.0.0.2

Pinging 20.0.0.2 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127
Reply from 20.0.0.2: bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>ping 20.0.0.3

Pinging 20.0.0.3 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.3: bytes=32 time=0ms TTL=127
Reply from 20.0.0.3: bytes=32 time=0ms TTL=127
Reply from 20.0.0.3: bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>
```



WEEK 5

Configure Web Server, DNS within a LAN.
OBSERVATION:



20/07/23

~~Output~~ Configure web server, DNS within a LAN

① ★ Create topology -

Fa0/1 Fa 1/1 switch Fa0 Fa0 PC server

★ Steps -

① Create topology as shown in the figure.
② let the PC ip address as 10.0.0.10
③ let the server ip address as 10.0.0.20
④ Turn on the DNS in services and add name and address same as ip address.
⑤ In HTTP go to index.html and using the normal html commands add name and UIN.
<h1> Ria Jain </H1>
<h1> 1BM21CS163 </H1>
⑥ Go to web browser in PC and enter the URL same as name of website you saved in server.

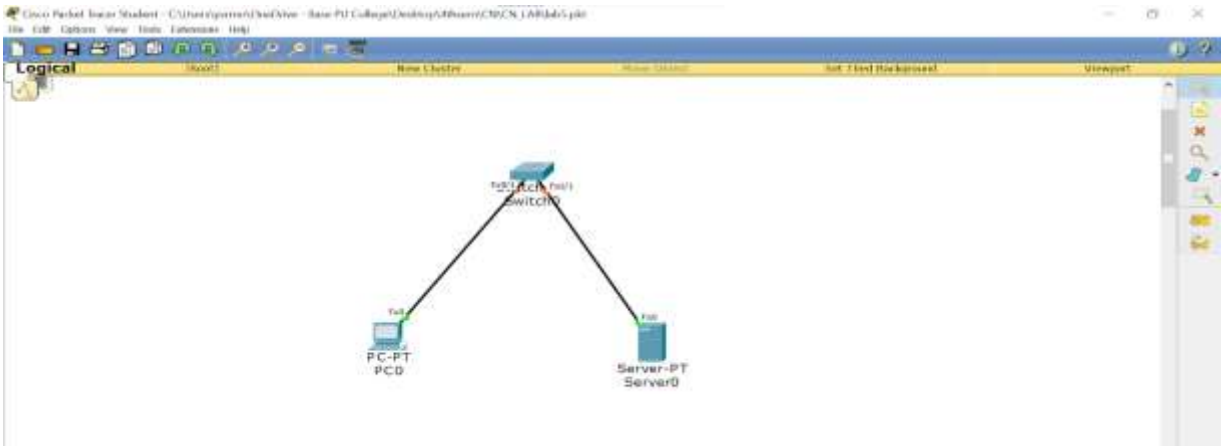
★ Output -

In web browser in PC we can now enter the name of server and we are shown the index.html page.

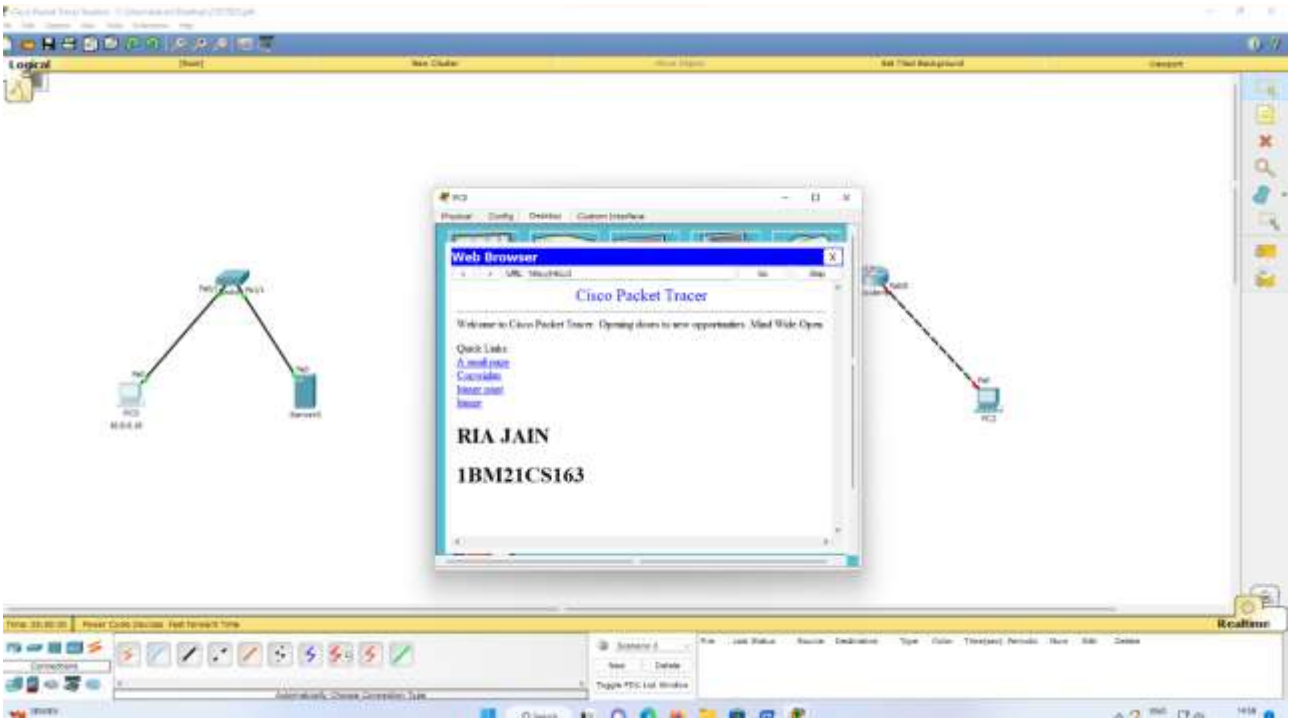
Ria Jain
1BM21CS163

web browser
url http://example.com
cisco packet tracer
Ria Jain
1BM21CS163

TOPOLOGY:



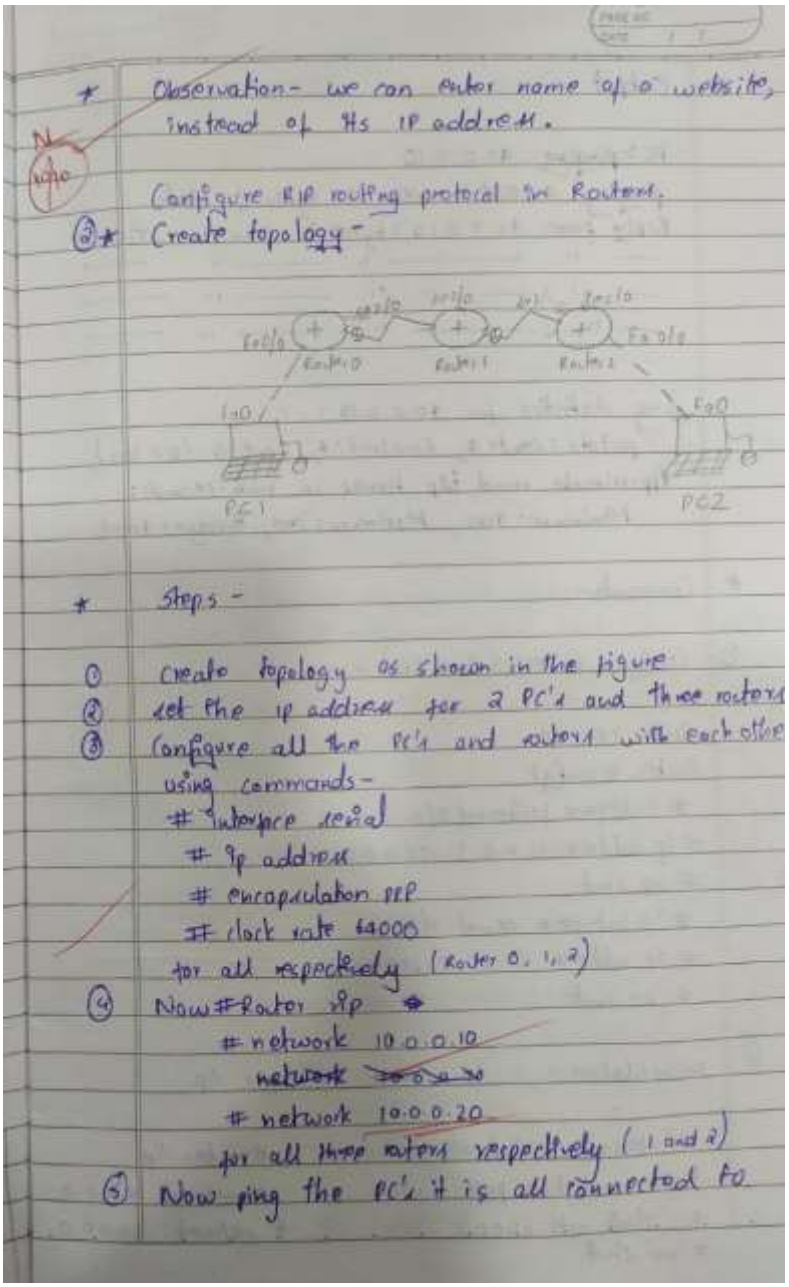
OUTPUT:



WEEK 6

Configure RIP routing Protocol in Routers.

OBSERVATION:



* Observation- we can enter name of a website, instead of its IP address.

Configure RIP routing protocol in Routers.

② * Create topology -

Fa0/0 Router 0 — Router 1 — Router 2 Fa0/0

10.0 PC1 Fa0 PC2

* Steps -

① Create topology as shown in the figure
② set the IP address for 2 PC's and three routers
③ Configure all the PC's and routers with each other using commands -
interface serial
ip address
encapsulation PPP
clock rate 64000
for all respectively (Router 0, 1, 2)
④ Now # Router rip
network 10.0.0.10
~~network 10.0.0.20~~
network 10.0.0.20
for all three routers respectively (1 and 2)
⑤ Now ping the PC's it is all connected to

PAGE NO.
DATE / /

* Output -

PC > pinging 40.0.0.10
pinging 40.0.0.10 with 32 bytes of data:
Reply from 40.0.0.10 : bytes=32 time=4ms 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 = 4ms, Maximum = 7ms, Average = 4ms

* Commands -

① Configure Routers (plan ips) -

9/10

Router > enable
Router # config t
interface fastthernet 0/0
ip address 10.0.0.3 255.0.0.0
no shut
ip interface serial 2/0
ip address 20.0.0.1 255.0.0.0
no shut

23/7/23

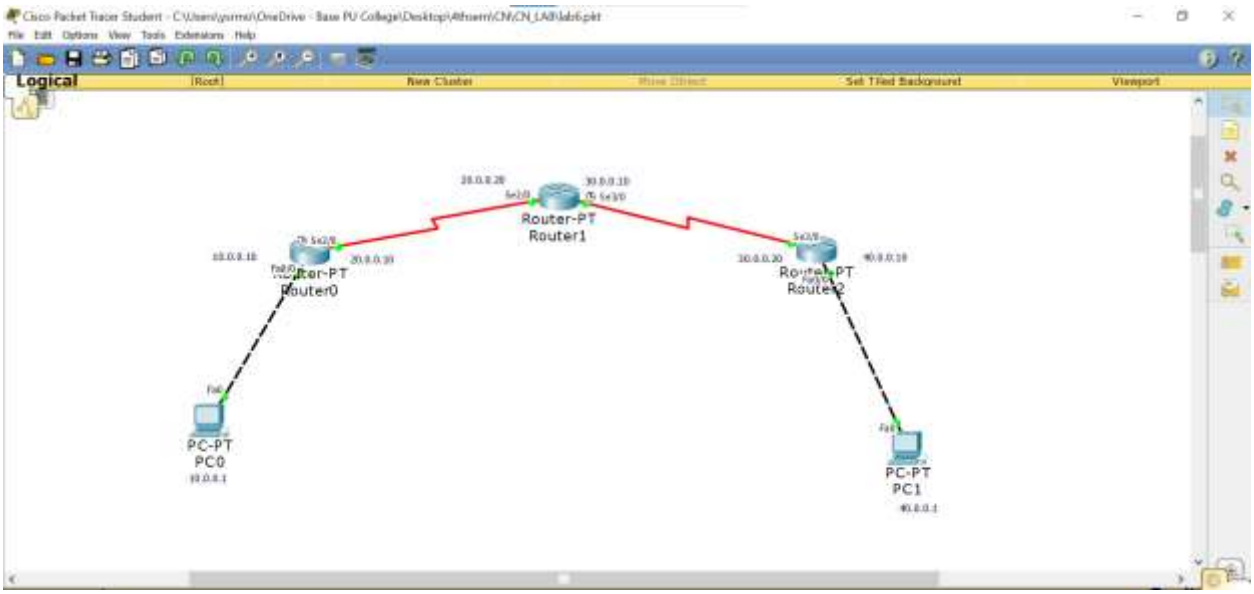
② encapsulation -

interface serial 2/0
encapsulation ppp
clock rate 64000
no shut

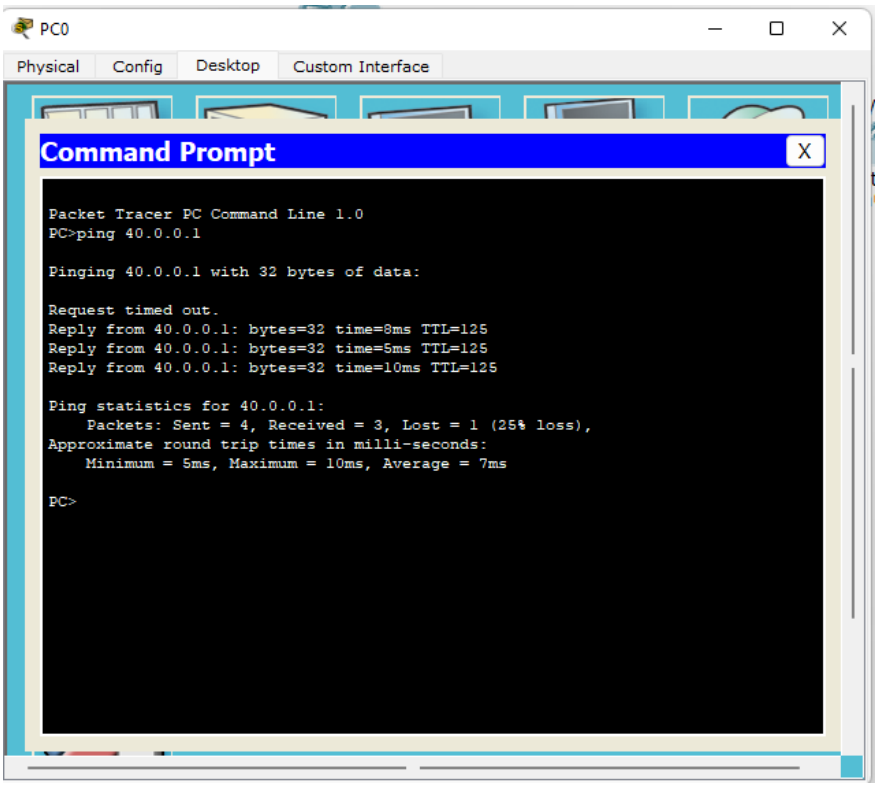
③ for rip.

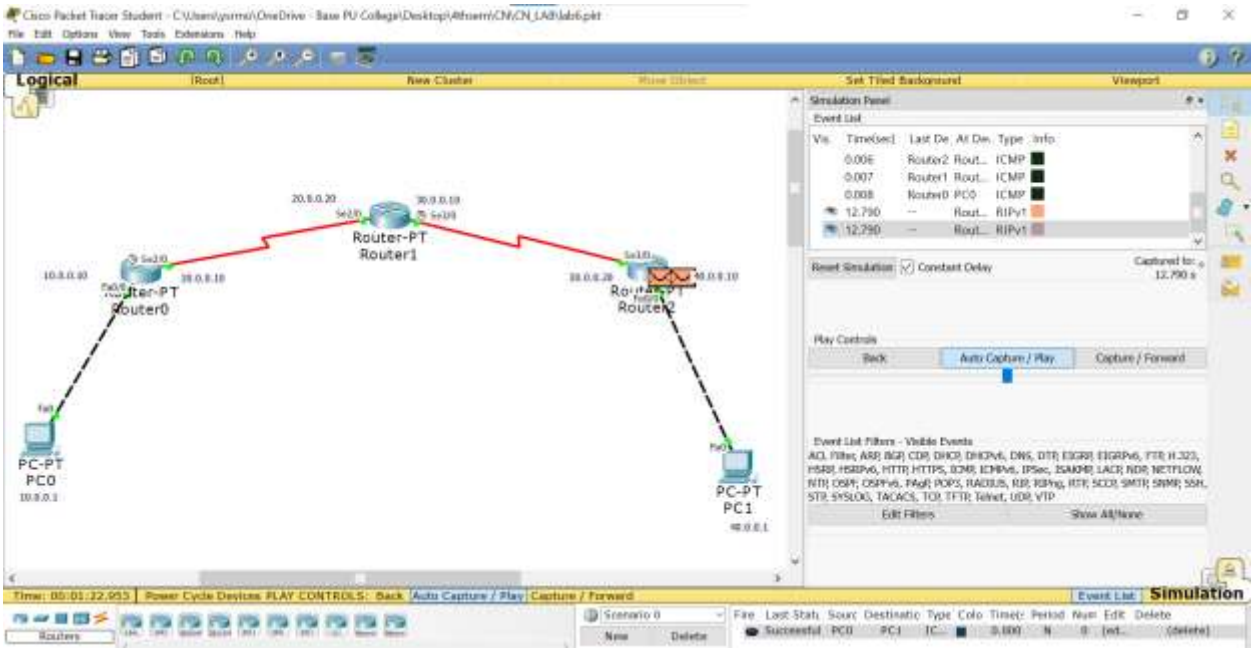
router rip
network 10.0.0.0
network 20.0.0.0

TOPOLOGY:



OUTPUT:

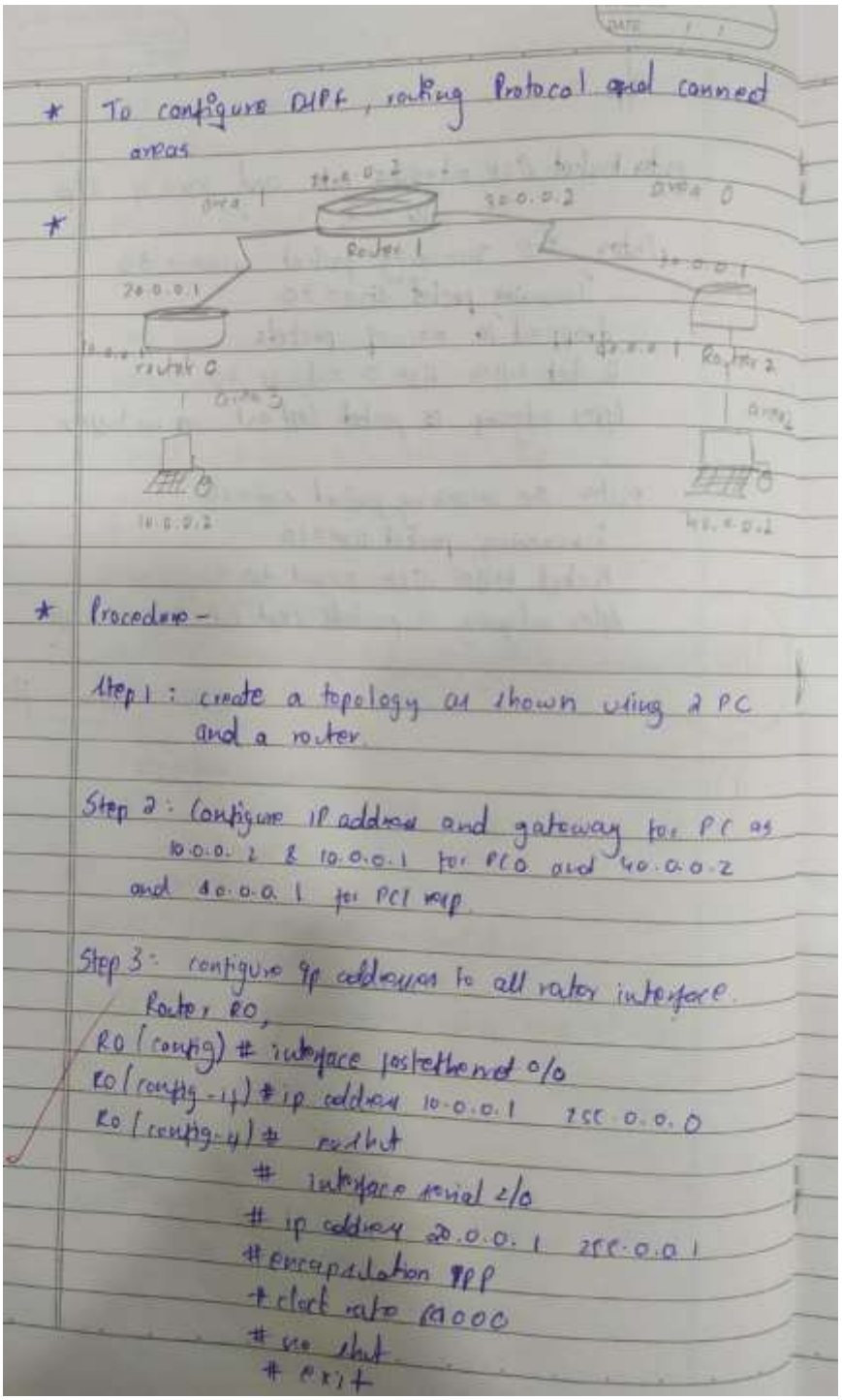



Logical
Router-PT
Router1
Router-PT
Router0
Router-PT
Router2
PC-PT
PC0
PC-PT
PC1
Simulation Panel
Event List
Play Controls
Back
Auto Capture / Play
Capture / Forward
Simulation

WEEK 7

Configure OSPF routing protocol.

OBSERVATION:



* To configure OSPF, routing Protocol and connect areas

* Procedure -

Step 1 : create a topology as shown using 2 PC and a router.

Step 2 : Configure IP address and gateway for PC as 10.0.0.2 & 10.0.0.1 for PC0 and 40.0.0.2 and 40.0.0.1 for PC1 resp.

Step 3 : configure ip address to all router interface.

Router R0,

R0 (config) # interface fastethernet 0/0
R0 (config-if) # ip address 10.0.0.1 255.0.0.0
R0 (config-if) # no shut
interface serial 2/0
ip address 20.0.0.1 255.0.0.1
encapsulation PPP
clock rate 64000
no shut
exit

Adora
PAGE NO.
DATE / /

IIIly for R1 and R2

Step 4 : Now, enable ip routing by configuring OSPF routing protocol.

Router R0,

```
# rater ospf 1
# rocter id 1.1.1.1
# network 10.0.0.0 0.255.255.255 area 3
# network 20.0.0.0 0.255.255.255 area 1
# exit
```

IIIly for R1 & R3

```
# router ospf 2
# rocter id 2.2.2.2 area 0
# network 30.0.0.0 255.255.255
# network 40.0.0.0 255.255.255
```

Step 5 : Now check routing table for route # show ip route.

C - connected
O - OSPF
C - 10.0.0.0/8 is directly connected, Fa0/0
C 20.0.0.0/8 —— " —— Fe2/0
O. IA 40.0.0.0/8 via 20.0.0.2, 00:04:23 Fa0/0
IA 30.0.0.0/8 via 20.0.0.2, 00:07:29.

Here R1 known area 0. Network 20.0.0.0 connected to R1 from R0, R0 learns network through

```
# router ospf 1
```

⇒ process id (1-65535)

~~There cannot the same interface~~

```
R0(config-if) # interface loopback 0
R0(config-if) # ip add 172.16.1.252 255.255.0.0
```

DATE / /

Step 6: Now check routing table for R3
R3 # show ip route
codes : O - ospf, C - connected
O IA 20.0.0.0/8 via 30.0.0.2,
00:18:58, se 2/0
C 40.0.0.0/8 directly connected,
fastethernet 0/0
C 30.0.0.0/8 is directly connected, se 2/0

Step 7: create virtual link b/w R0, R1 by that
we create a virtual link in area 380.

In R0
R0 (config) # router ospf 1
R0 (config-router) # area 1 virtual
link 2.2.2.2

In R1
R1 (config) # router ospf 1
R1 (config-router) # area 1 virtual
link 1.1.1.1

Step 8: R1 & R2 get updates about area 5.
Now check routing table for R2.

R2 # show ip route
Codes : O - Ospf C - connected
O IA 20.0.0.0/8 via 30.0.0.2, 00:01:05 se 2/0
C 40.0.0.0/8 is directly connected
fastethernet 0/0
O IA 10.0.0.0/8 via 30.0.0.2 00:01:56 se 2/0

Results. –

In PC0

PC > ping 40.0.0.2

pinging 40.0.0.2 with 32 bytes of data

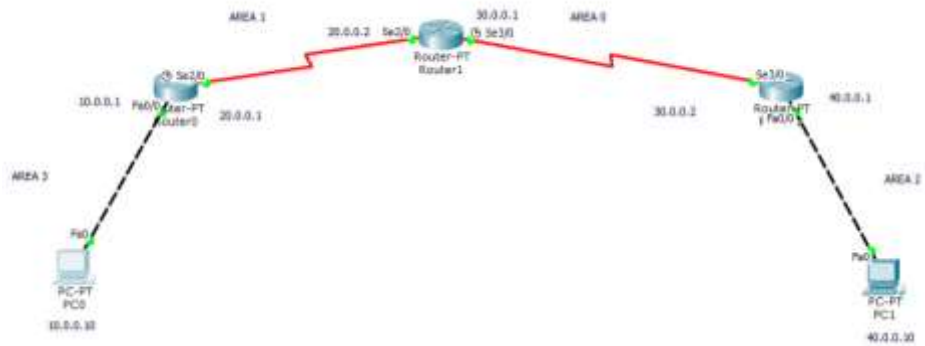
Reply from 40.0.0.2 : bytes = 32 time = 2ms
Reply from 40.0.0.2 : bytes = 32 time = 10 ms
Reply from 40.0.0.2 : bytes = 32 time = 14ms
Reply from 40.0.0.2 : bytes = 32 time = 2 ms
ping statistic for 40.0.0.2

packets : sent : 4, received = 4, lost = 0
approx round trip times in ms
min : 2ms, max = 14 ms, avg = 7ms

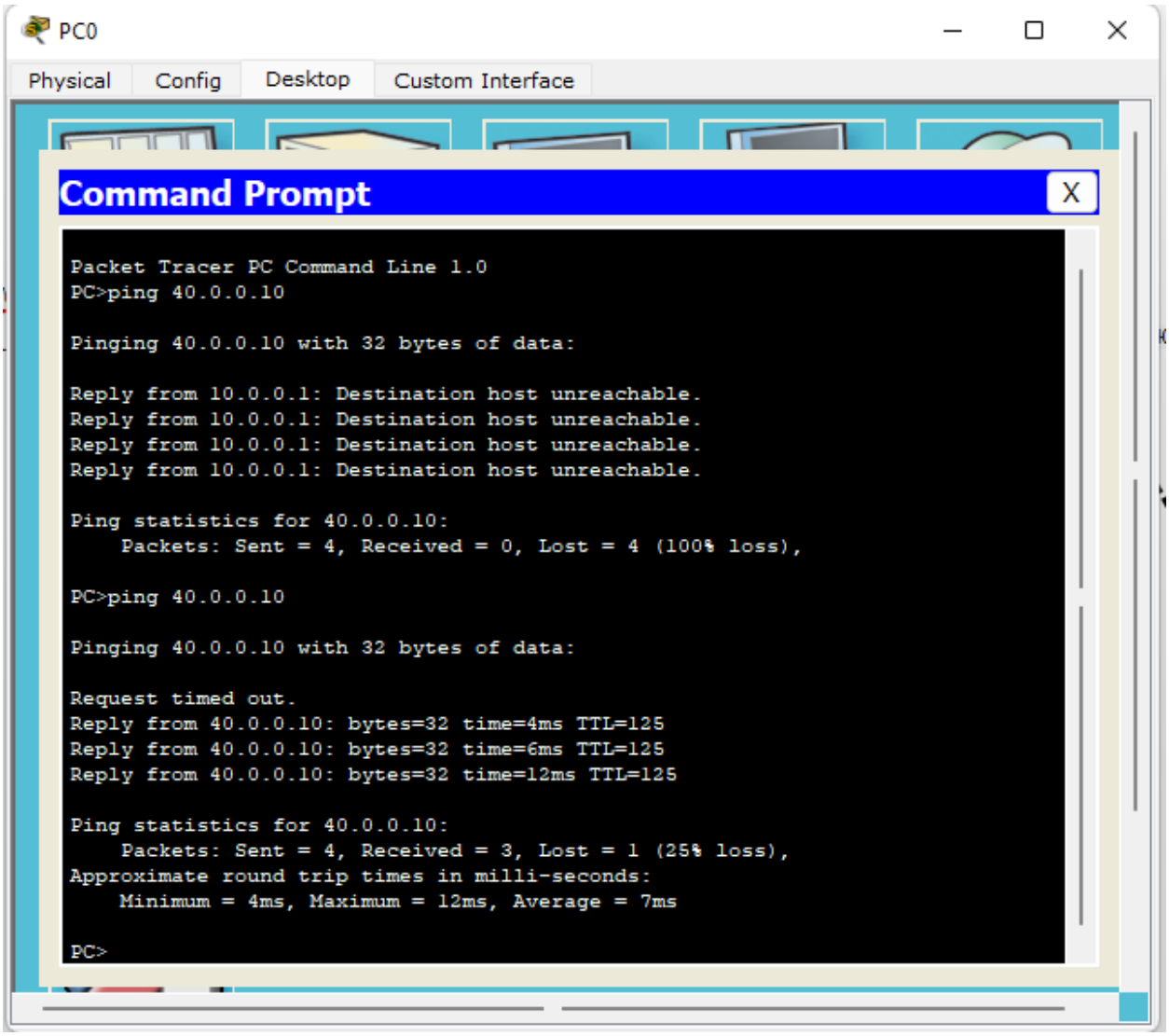
8/10

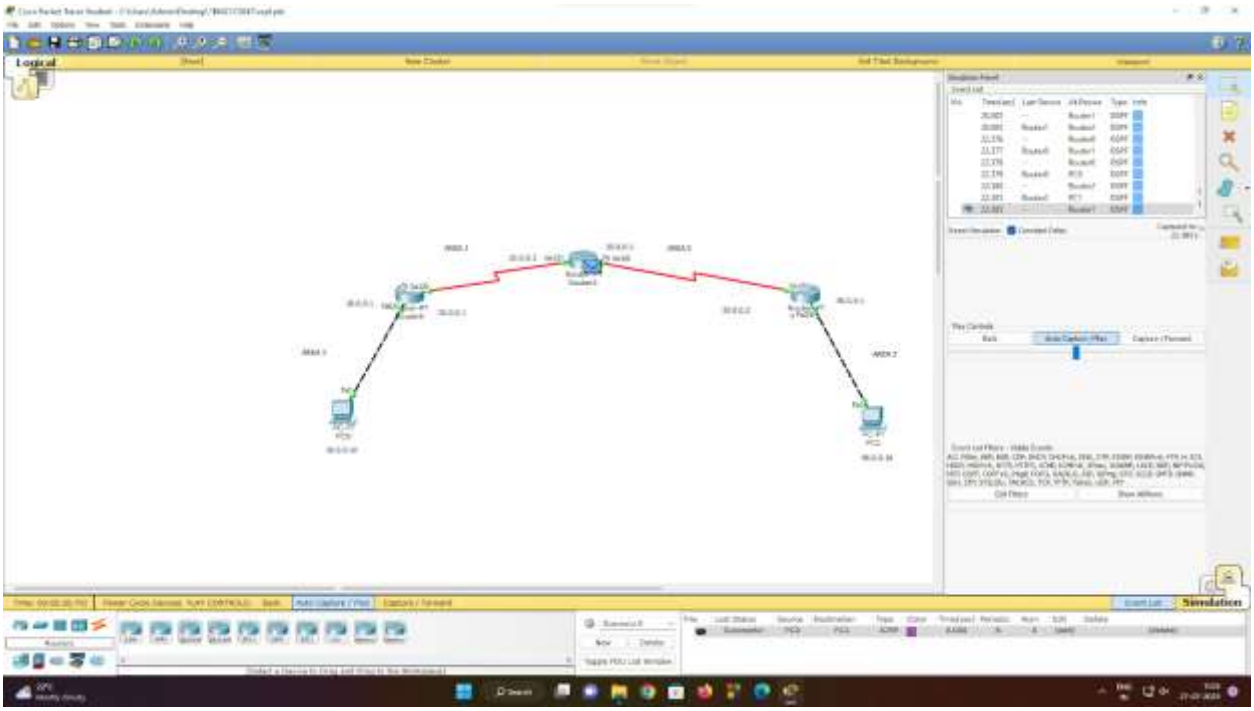
N
1/9/23

TOPOLOGY:



OUTPUT:

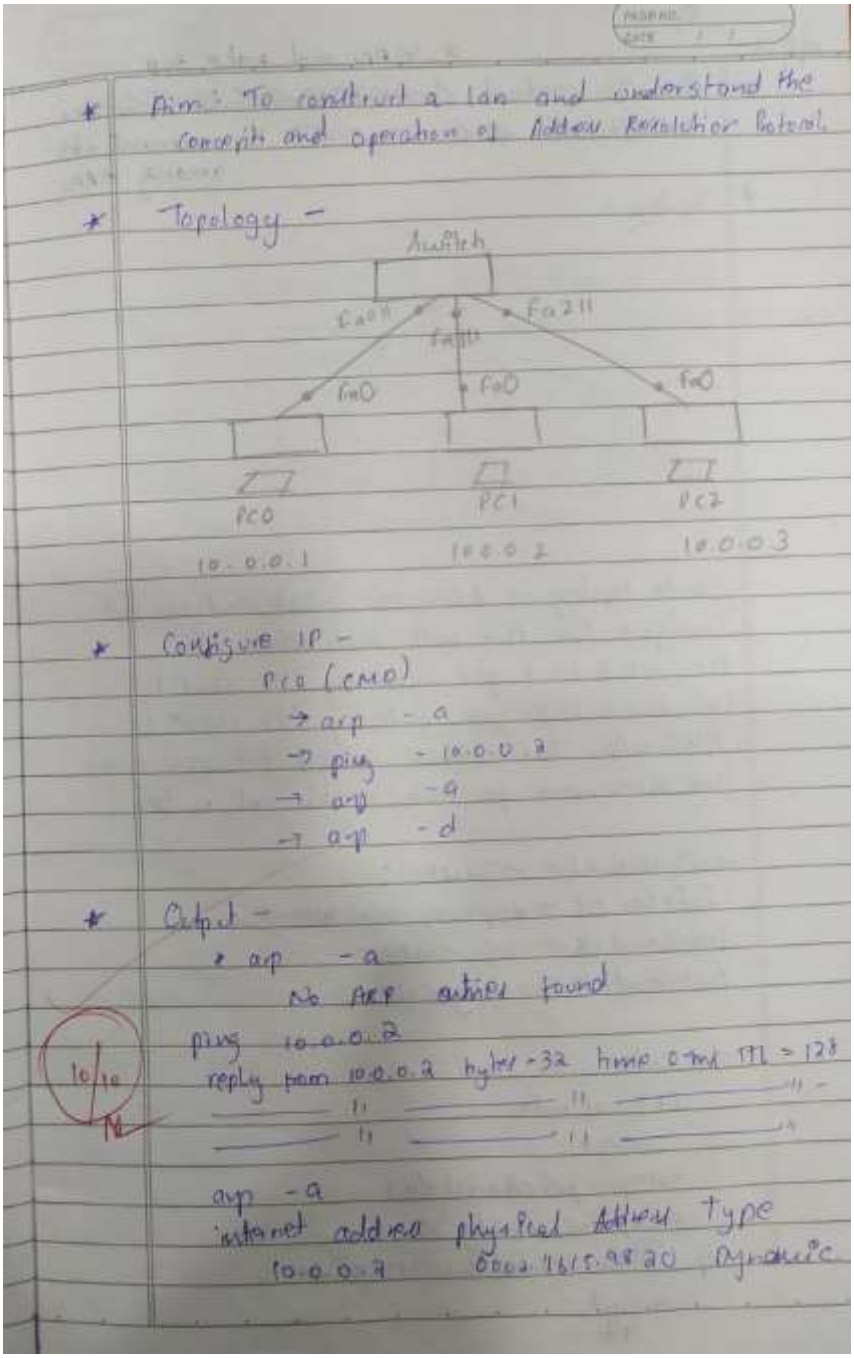



Logical
Simulation

WEEK 8

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

OBSERVATION:



* Aim: To construct a lan and understand the concepts and operation of Address Resolution Protocol

* Topology -

Switch

Fa0/1 Fa0/2 Fa0/3

Fa0 Fa0 Fa0

PC0 PC1 PC2

10.0.0.1 10.0.0.2 10.0.0.3

* 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

" " "

" " "

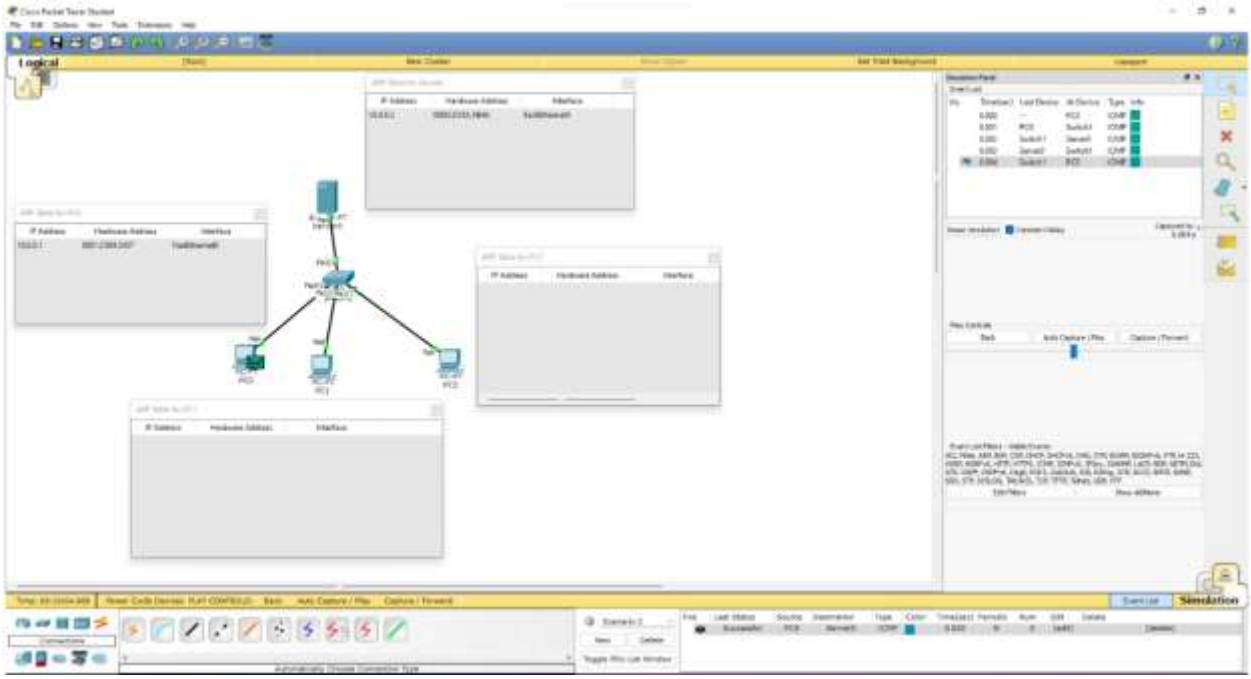
arp -a

internet address physical Address Type

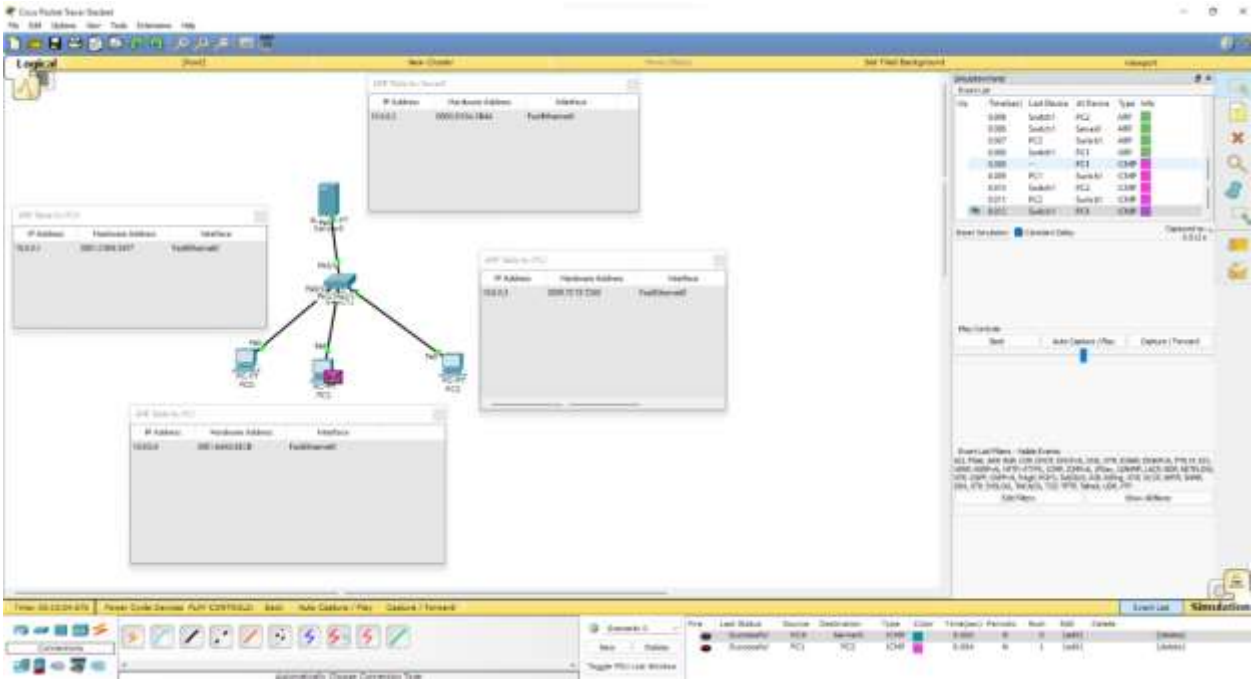
10.0.0.2 0002.1615.9820 Dynamic

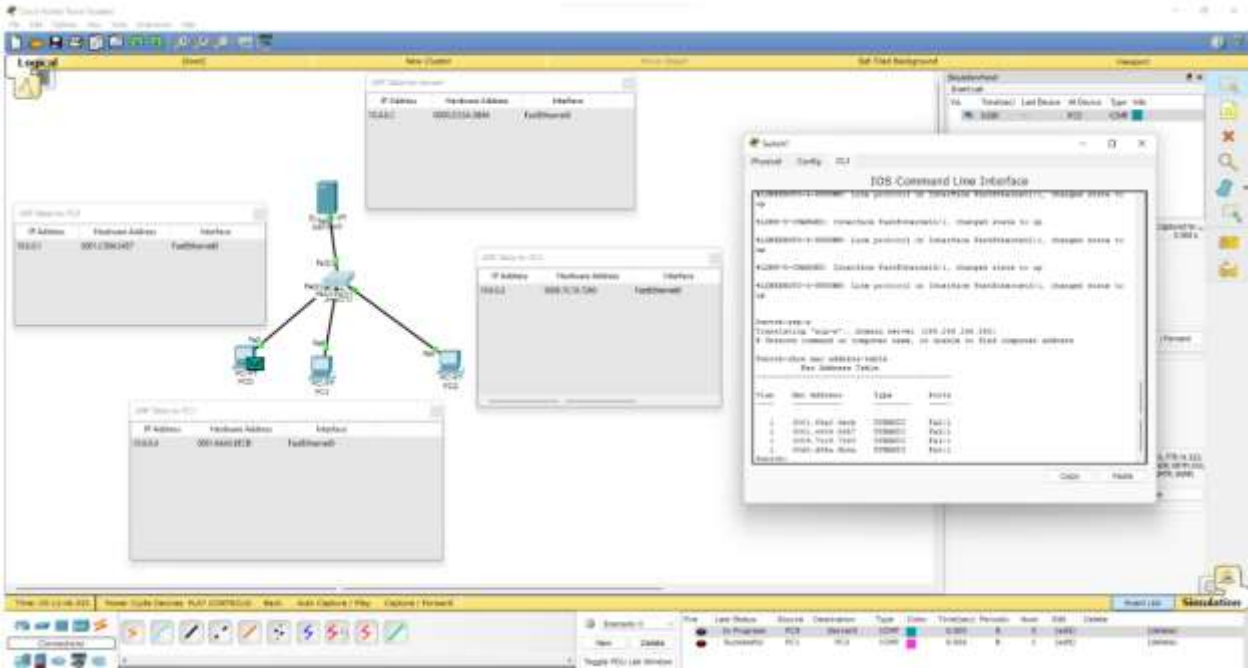
10/10

TOPOLOGY:



OUTPUT:

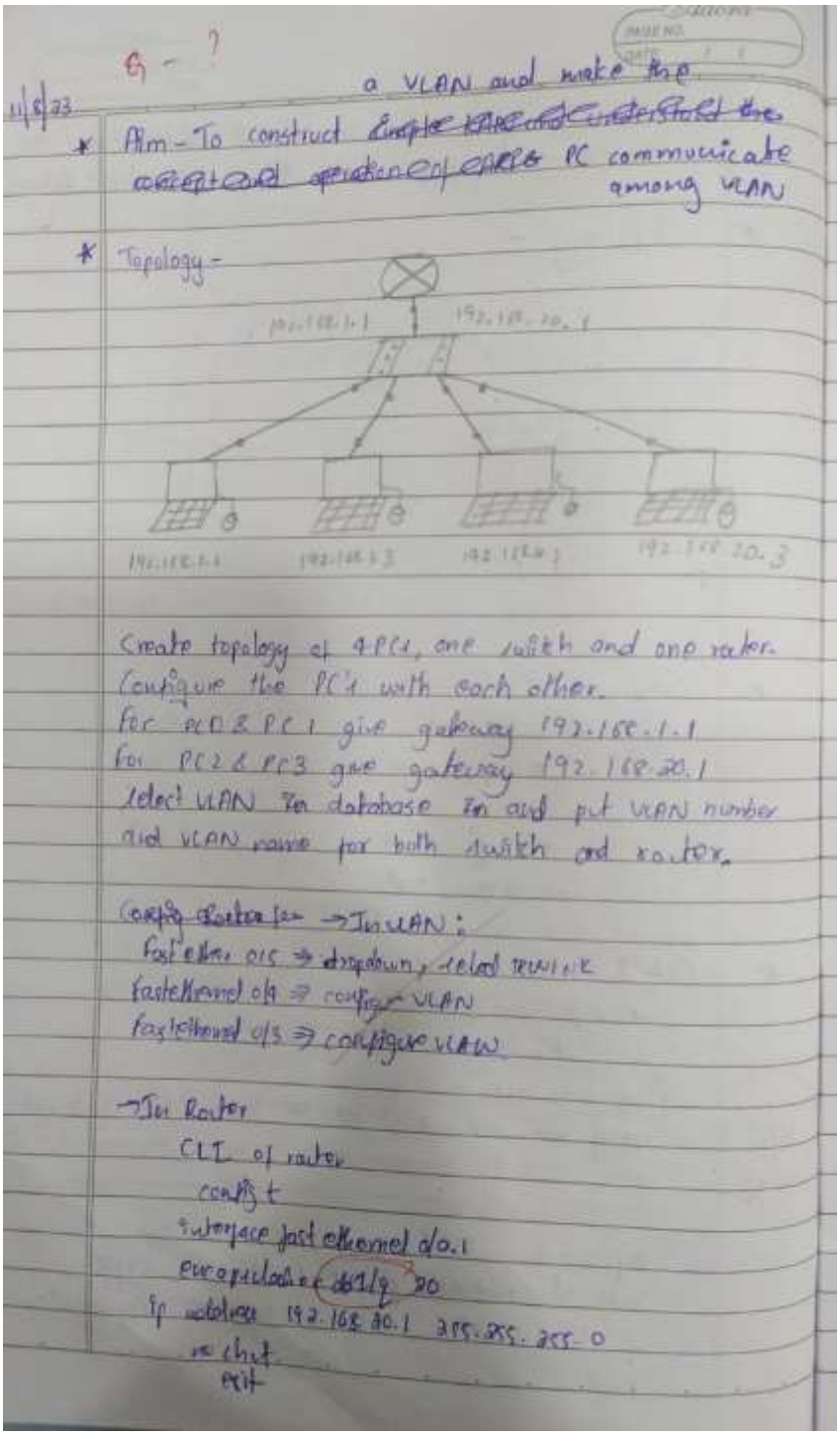



Logical
IOS Command Line Interface

WEEK 9

To construct a VLAN and make a pc communicate among VLAN.

OBSERVATION:



Q - ?

a VLAN and make the

11/8/23

* Aim - To construct ~~Simple LAN and understand the concept and operation of LAN~~ PC communicate among VLAN

* Topology -

192.168.1.1 192.168.20.1

192.168.1.1 192.168.1.3 192.168.20.2 192.168.20.3

Create topology of 4 PCs, one switch and one router.
Configure the PC's with each other.
For PC1 & PC1 give gateway 192.168.1.1
For PC2 & PC3 give gateway 192.168.20.1
Select VLAN in database in and put VLAN number and VLAN name for both switch and router.

~~Config Router for~~ → In VLAN:
Fastethernet 0/5 ⇒ dropdown, select TRUNK
fastethernet 0/4 ⇒ configure VLAN
fastethernet 0/3 ⇒ configure VLAN

→ In Router
CLI of router
config t
interface fast ethernet 0/0.1
encapsulation dot1q 20
ip address 192.168.20.1 255.255.255.0
no shut
exit

PAGE NO.
DATE / /

~~VLAN configure~~

★ Output -

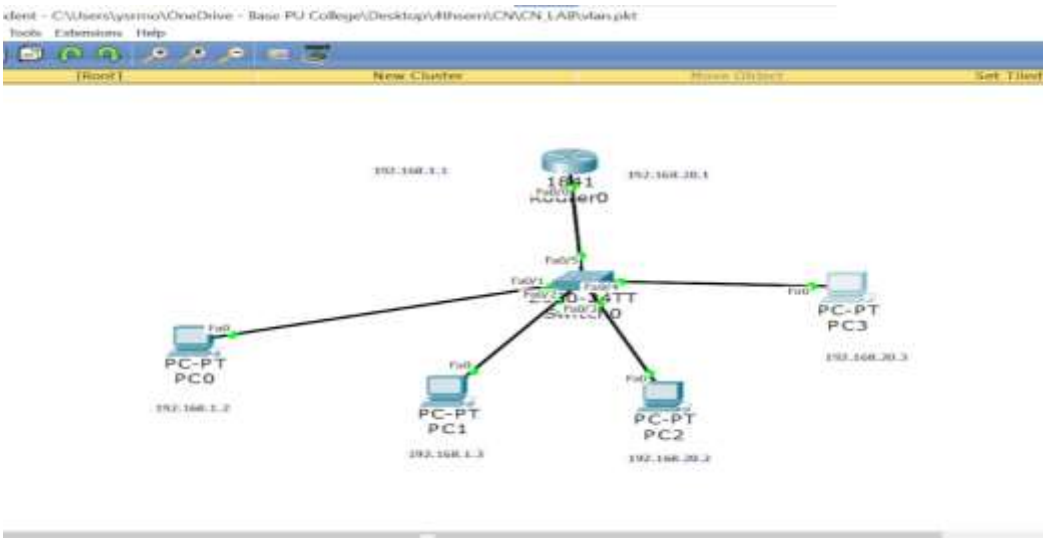
ping 192.168.20.3
pinging 192.168.20.3 with 32 bytes of data:
Reply from 192.168.20.3 : bytes=32 time=0ms TTL=127
" " " "
" " " "
" " " "

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

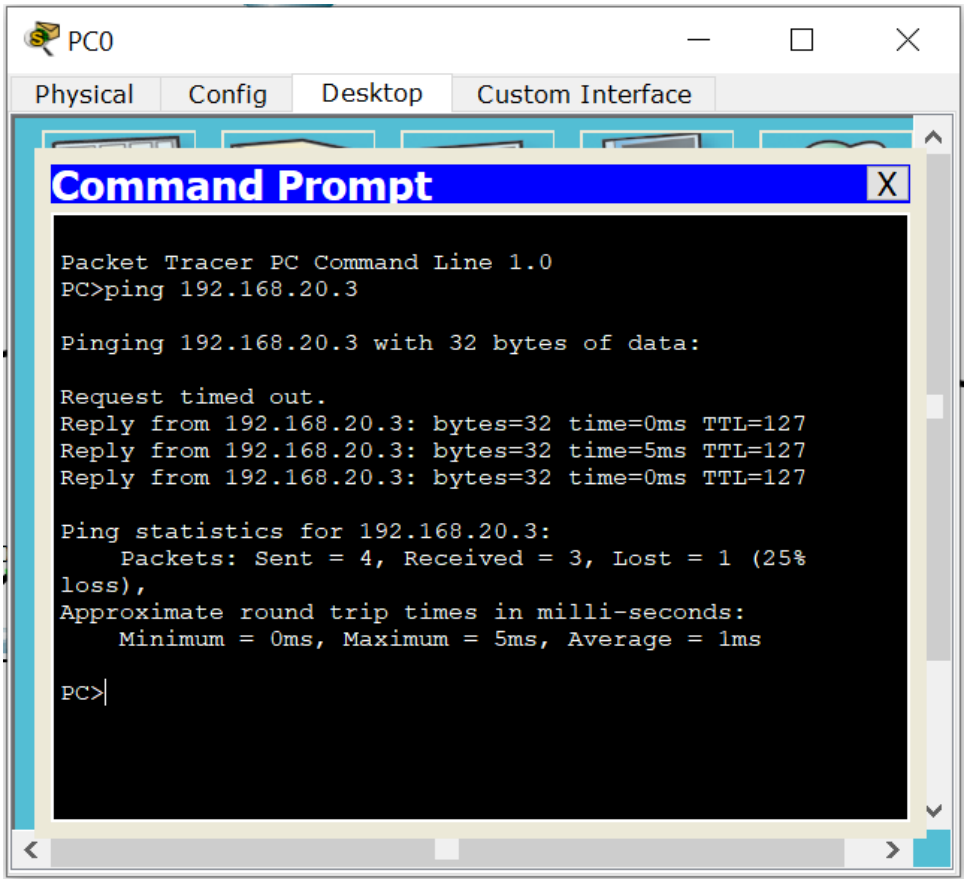
8/10

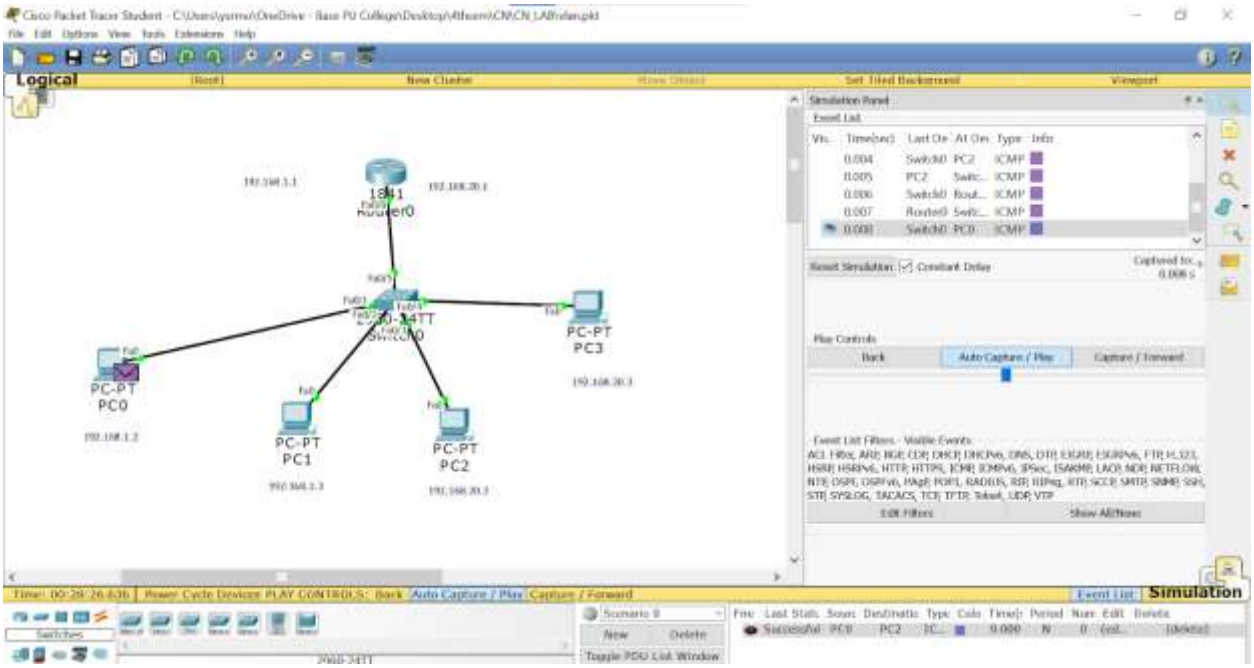
1/9/23

TOPOLOGY:



OUTPUT:

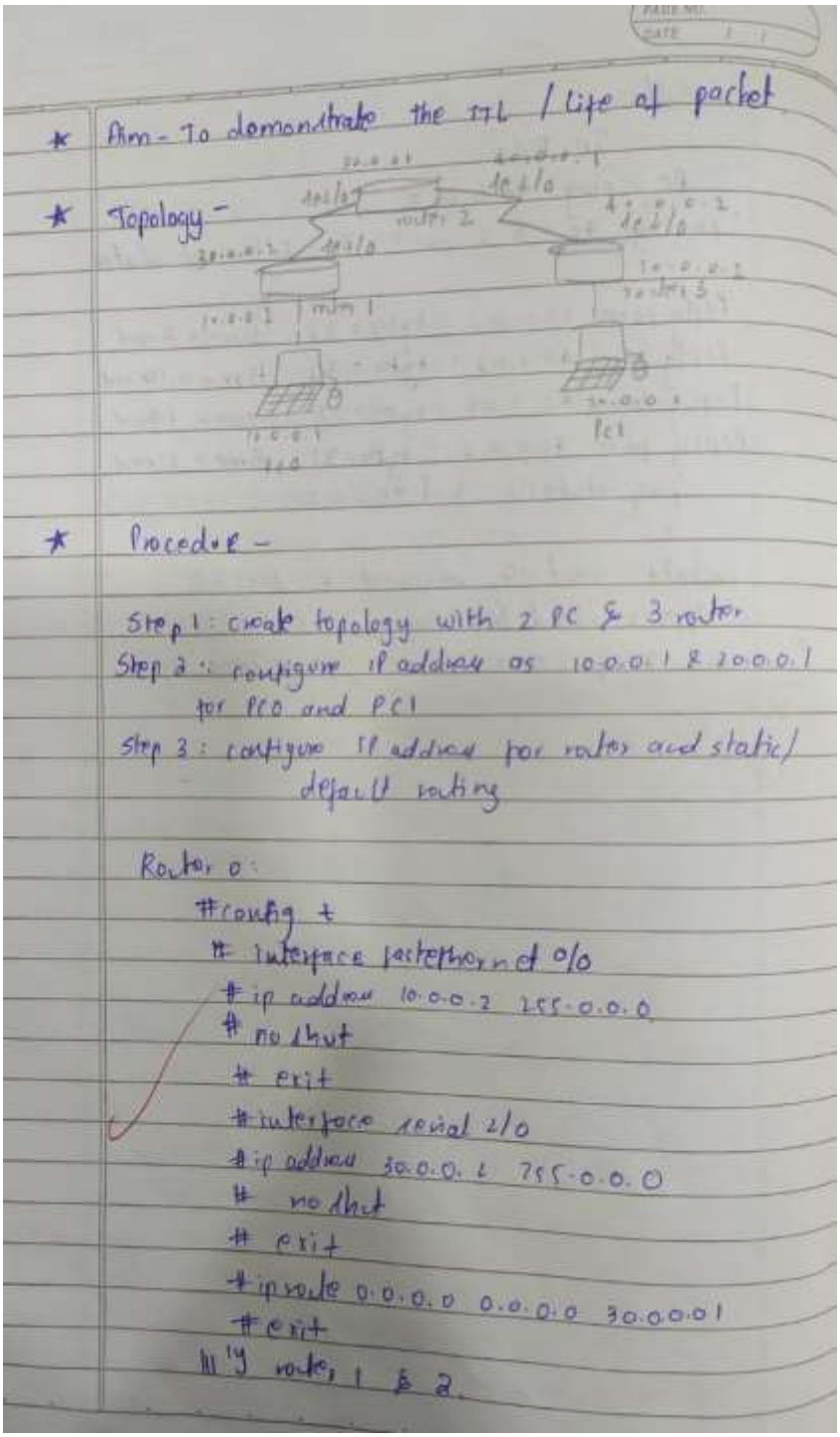



Logical
1841
Router0
PC-PT
PC0
PC-PT
PC1
PC-PT
PC2
PC-PT
PC3
Simulation

WEEK 10

Demonstrate the TTL/ Life of a Packet.

OBSERVATION:



* Aim - To demonstrate the TTL / Life of packet

* Topology -

* Procedure -

Step 1: create topology with 2 PC & 3 routers
Step 2: configure IP address as 10.0.0.1 & 20.0.0.1 for PC0 and PC1
Step 3: configure IP address for routers and static/default routing

Router 0:

```
#config t
# interface fastethernet 0/0
# ip address 10.0.0.2 255.0.0.0
# no shut
# exit
# interface serial 2/0
# ip address 30.0.0.1 255.0.0.0
# no shut
# exit
# ip route 0.0.0.0 0.0.0.0 30.0.0.1
# exit
```

III'y routers 1 & 2.

Step 4: in simulation mode, send a simple PDU from one PC to another.

Step 5: click on PDU during every transfer to see the inbound & outbound PDU details, use capture button to capture every transfer.

★ Observation:

1)

0	4	8	16	19	31
4	IHL	DSCP		TL: 28	
IP: 0x6			0x	0x0	
TTL: 255	PRO: 0x1			chksum	
SRC IP ID: 0.0.0.1					
DST IP 50.0.0.1					
OPT: 0x0				0x0	
Data (variable length)					

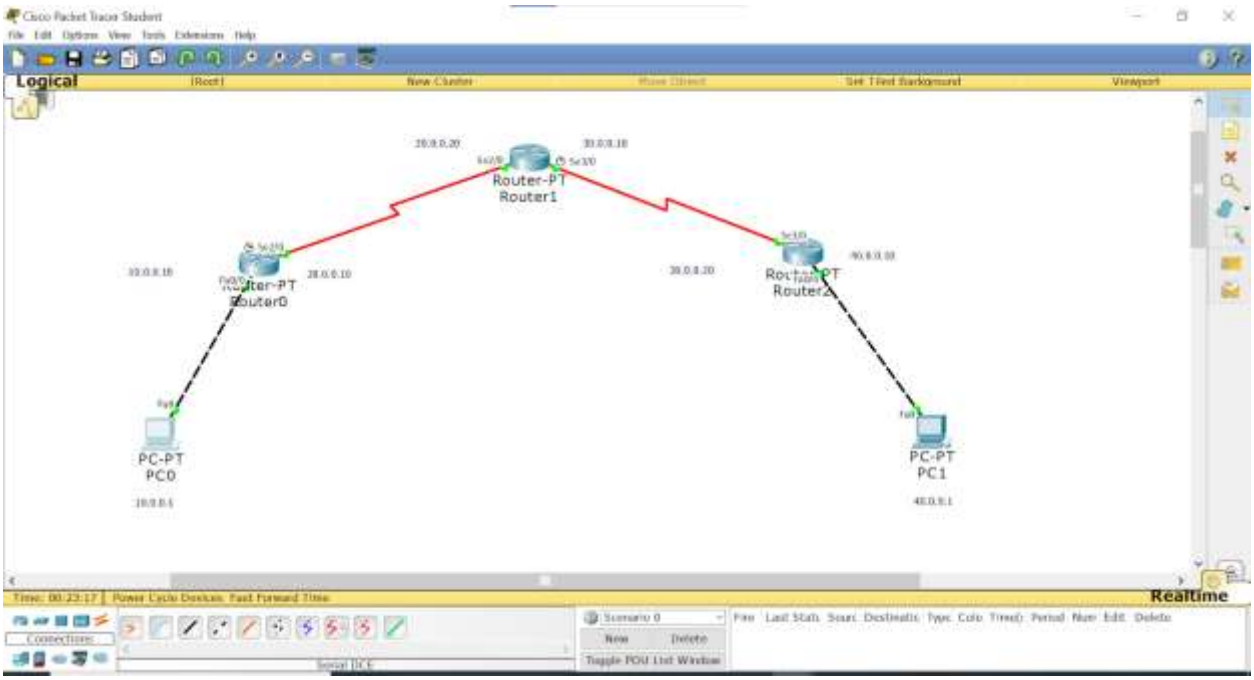
0	4	8	16	19	31
4	IHL	DSCP		TL: 28	
IP: 0x6	0x0			0x0	
TTL: 254	PRO: 0x1			chk1	
SRC IP: 10.0.0.1					
DST IP: 50.0.0.1					
OPT: 0x0				0x0	
Data (variable length).					

LS

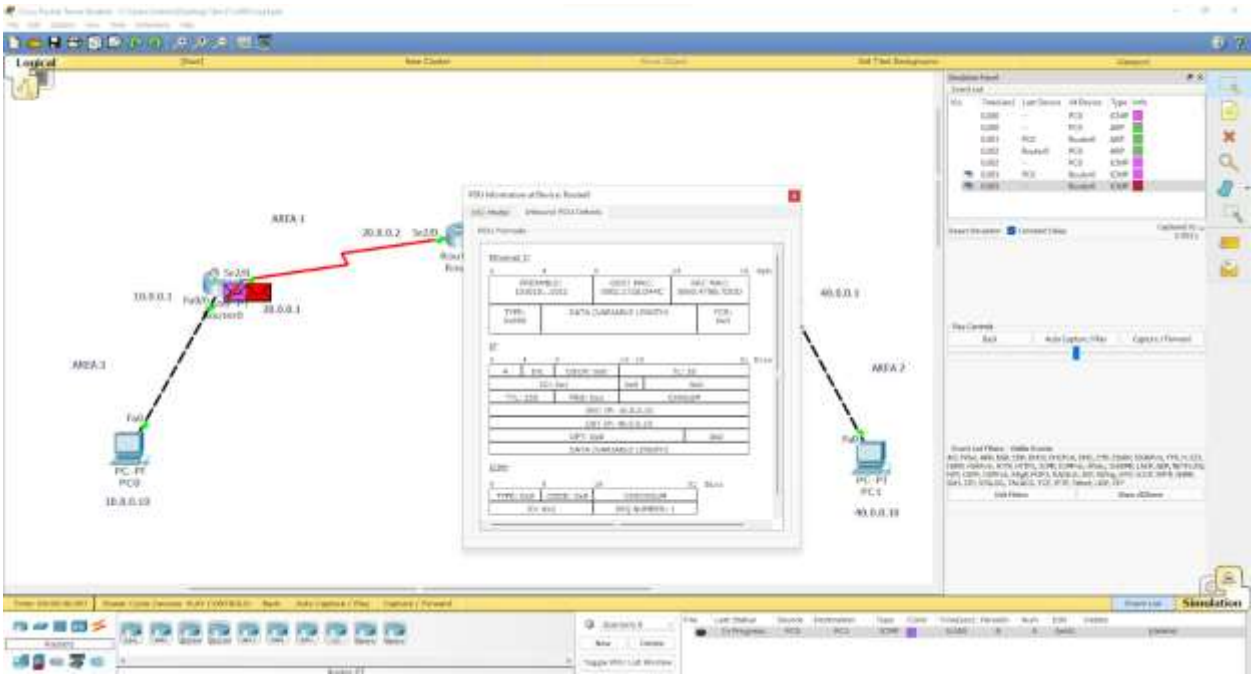
8/10

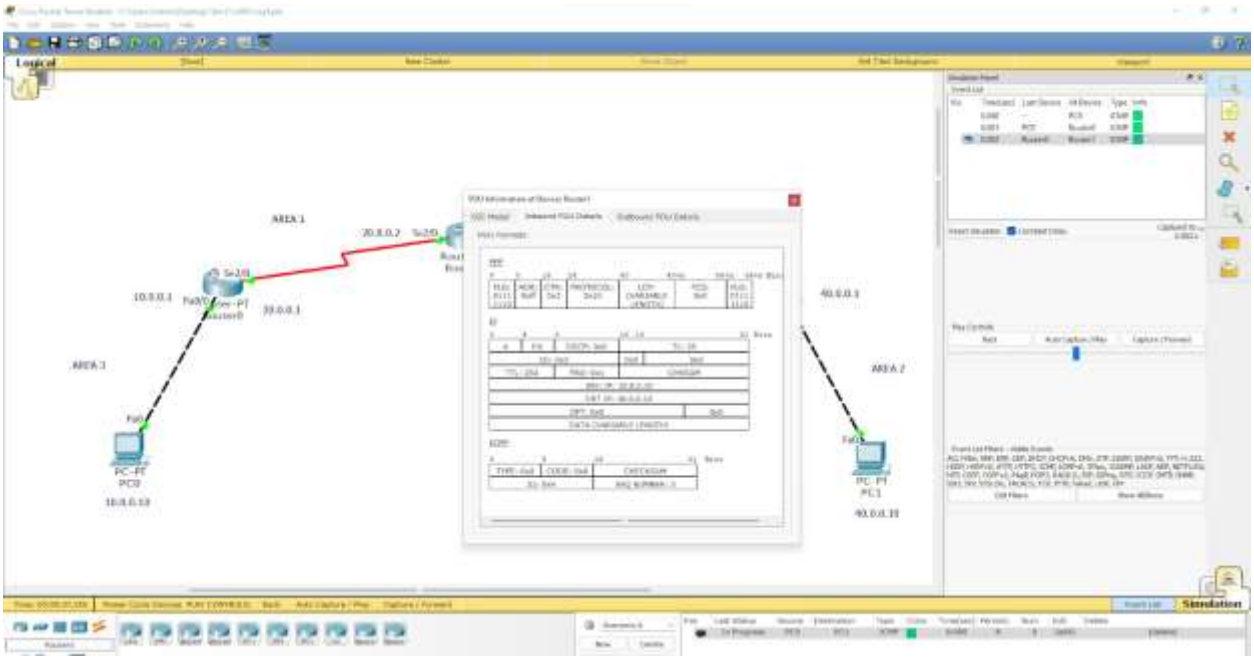
1/9/23

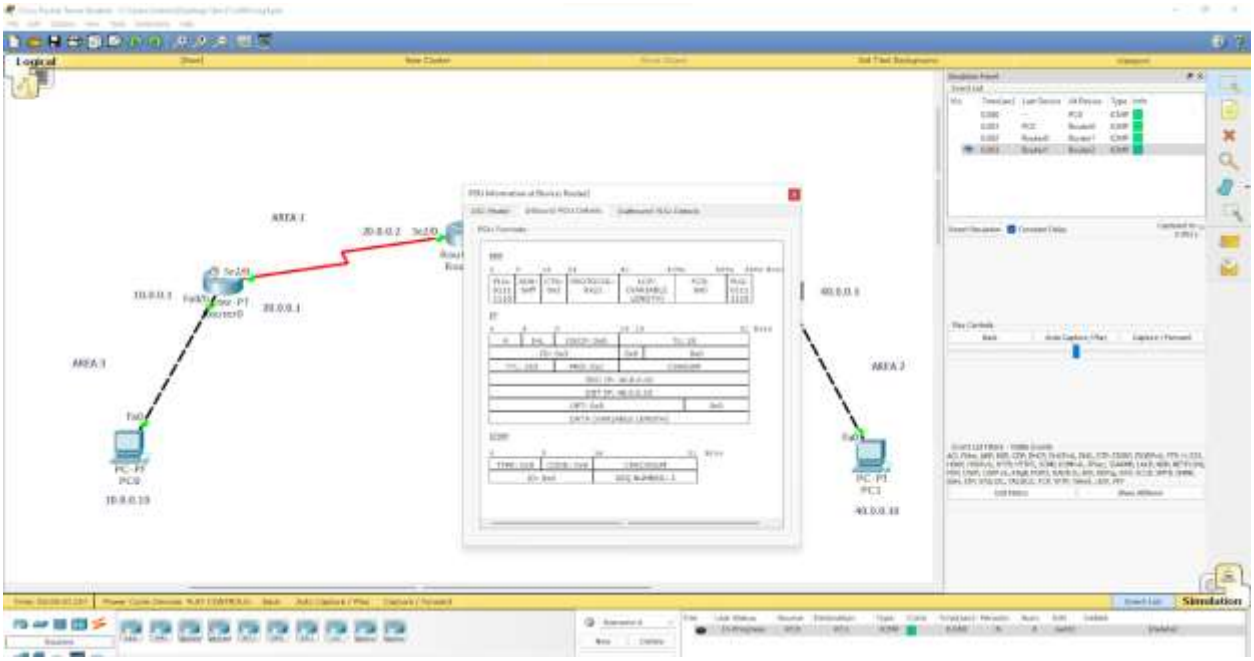
TOPOLOGY:

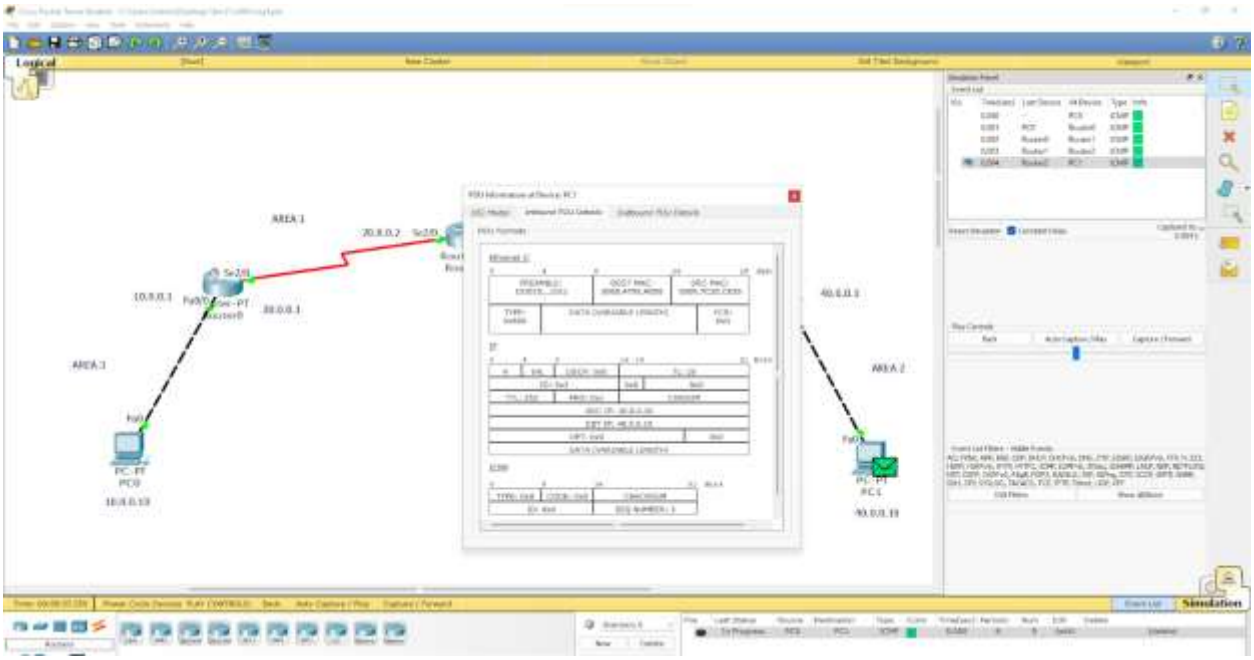


OUTPUT:




Logical
AREA 1
20.0.0.2
Se2/0
10.0.0.1
30.0.0.1
40.0.0.1
AREA 3
AREA 2
PC-PT
PC0
10.0.0.10
PC-PT
PC1
40.0.0.10
Simulation

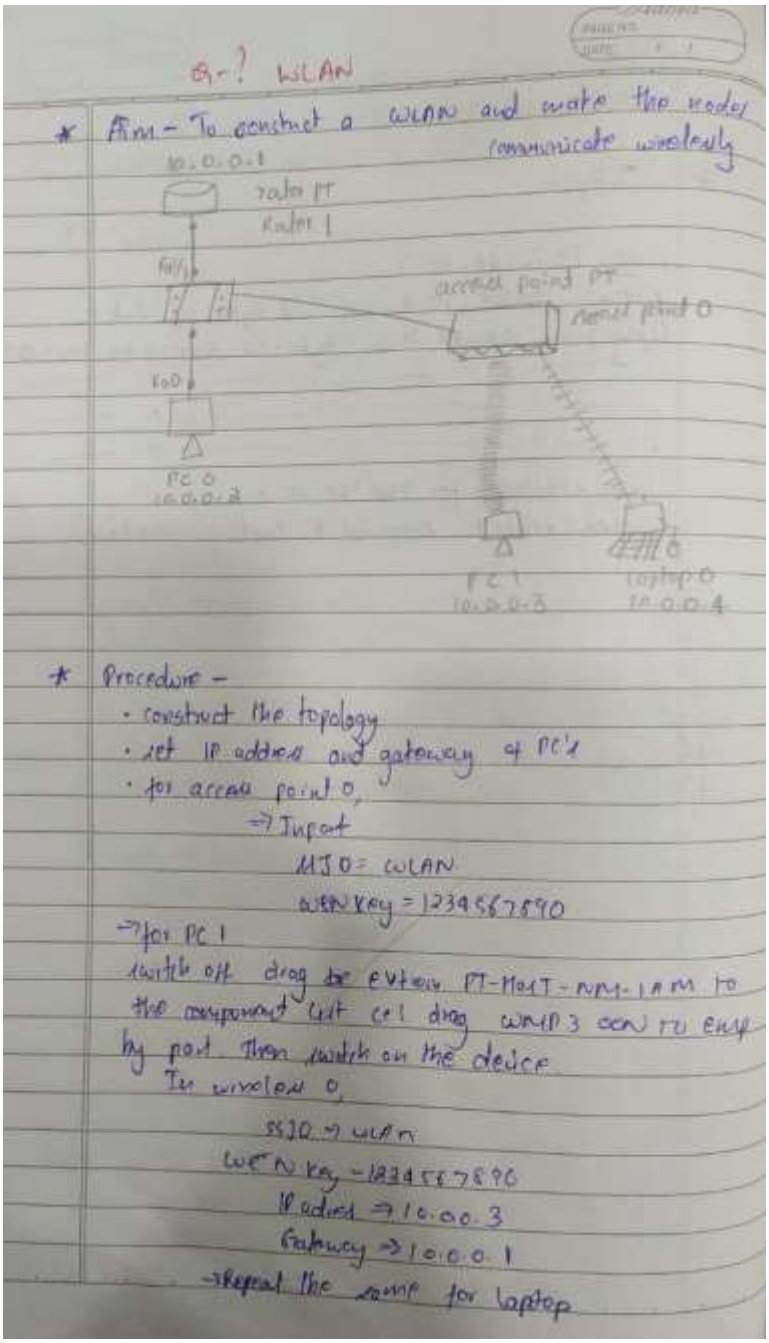

Logical
AREA 1
20.0.0.2
Se2/0
10.0.0.1
30.0.0.1
40.0.0.1
AREA 3
AREA 2
PC-PT
PC0
10.0.0.10
PC-PT
PC1
40.0.0.10
Simulation


Logical
AREA 1
20.0.0.2
Se2/0
10.0.0.1
30.0.0.1
AREA 3
Fa0/0
PC-PT
PC0
10.0.0.10
40.0.0.1
AREA 2
Fa0/0
PC-PT
PC1
40.0.0.10
Simulation

WEEK 11

To construct a WLAN and make the nodes communicate wirelessly

OBSERVATION:



PAGE NO.
DATE / /

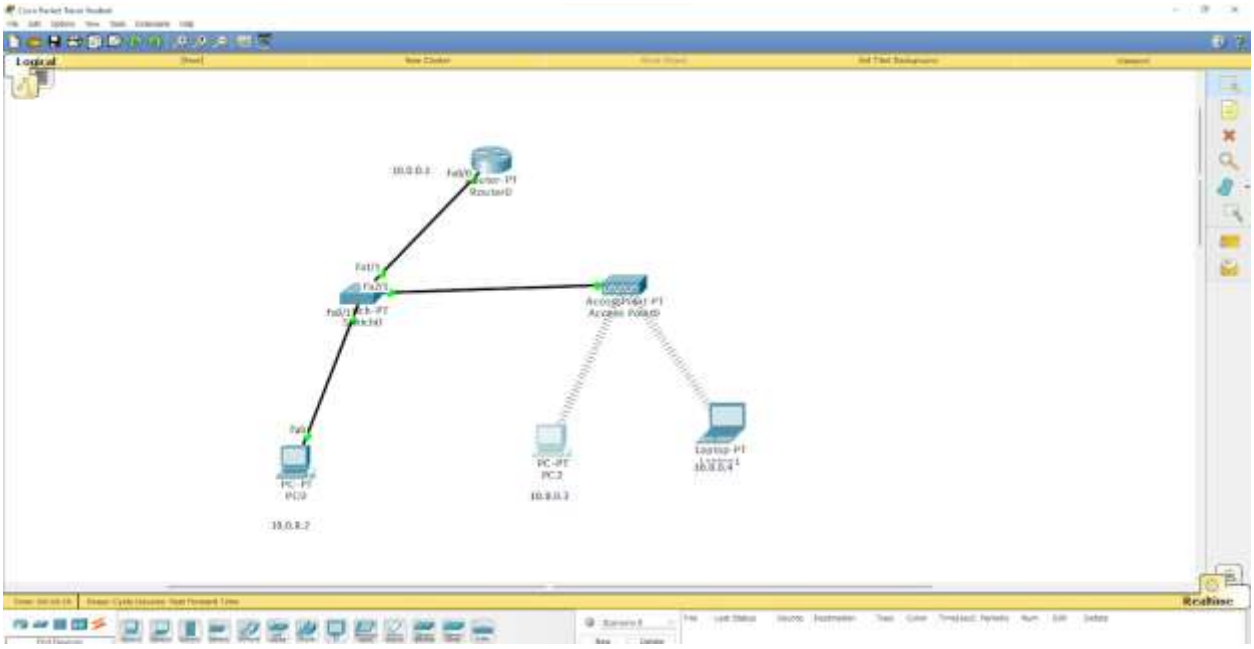
★ Output -

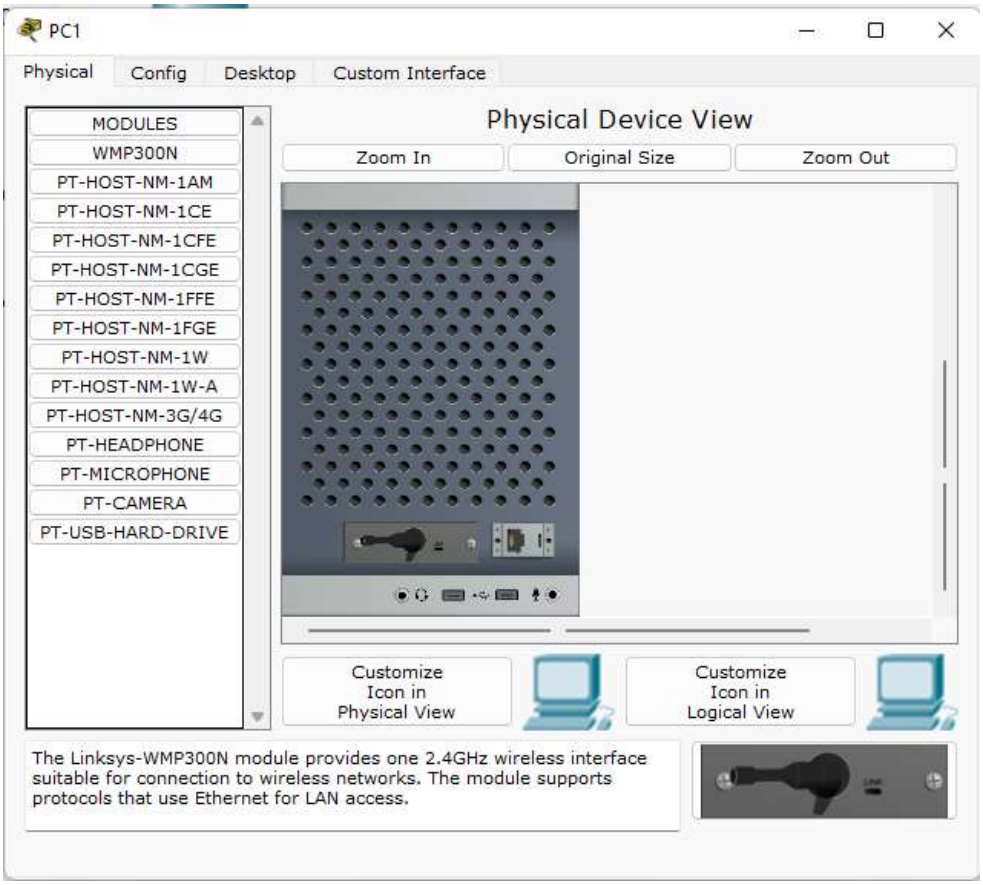
ping 10.0.0.3
pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3 : bytes = 32 bytes time=15ms TTL:126
" " " "
" " " "
" " " "

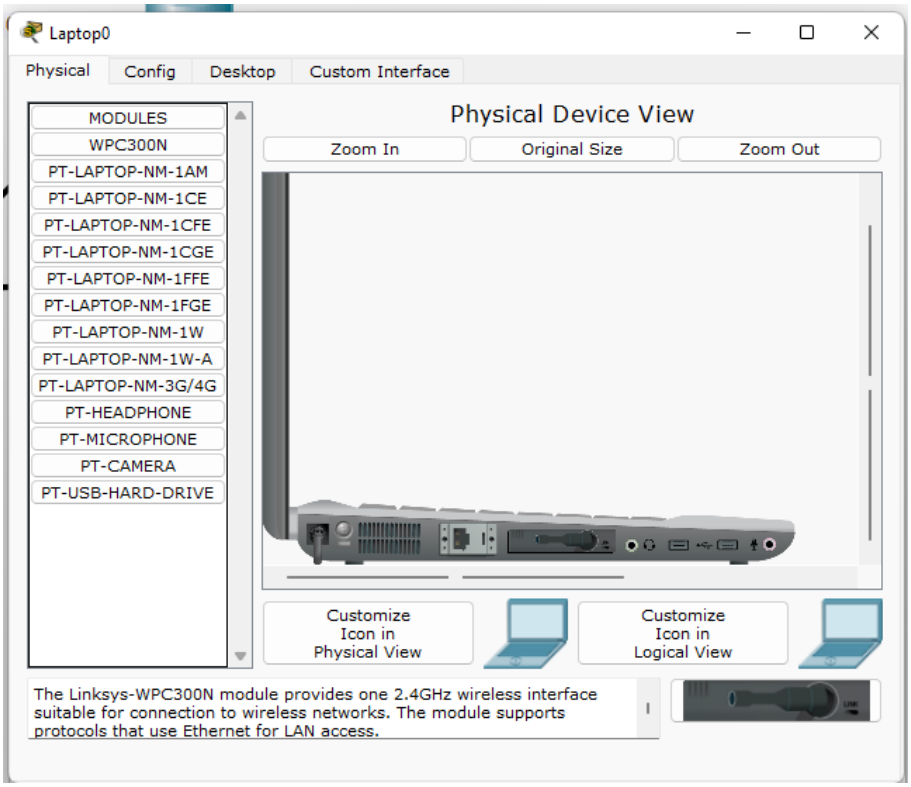
ping statistics for 10.0.0.3:
packet: sent=4, Received=4, lost=0
Approximate round trip time in millisecondes:
minimum: 6ms, maximum = 15ms, average 10ms

9/10

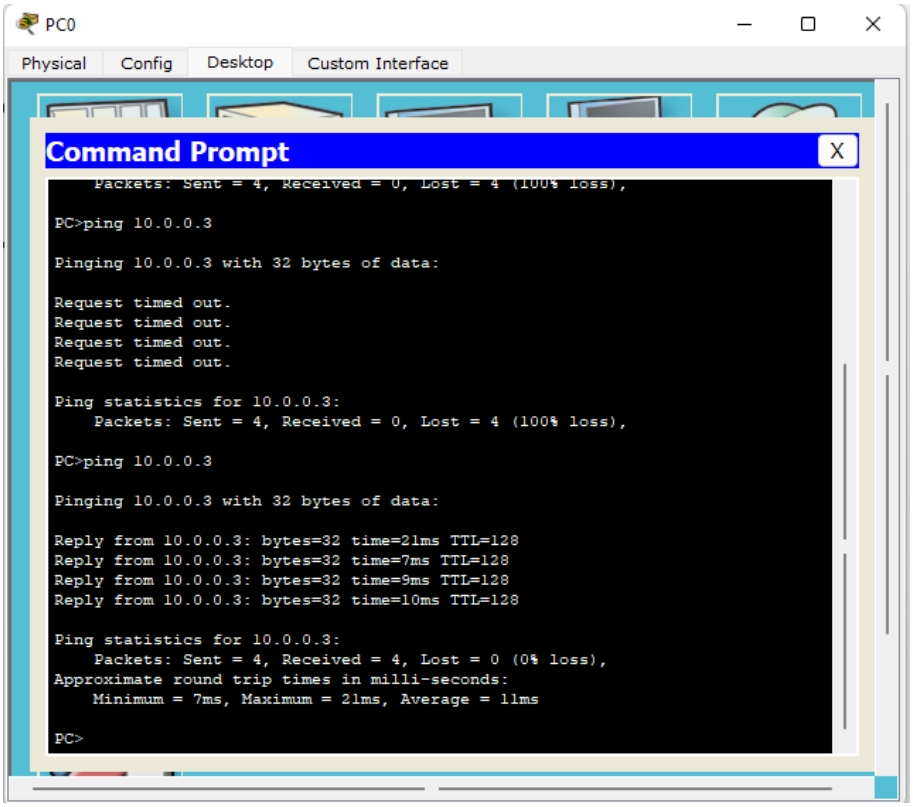
TOPOLOGY:




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.


Laptop0
Physical
Config
Desktop
Custom Interface
MODULES
WPC300N
PT-LAPTOP-NM-1AM
PT-LAPTOP-NM-1CE
PT-LAPTOP-NM-1CFE
PT-LAPTOP-NM-1CGE
PT-LAPTOP-NM-1FFE
PT-LAPTOP-NM-1FGE
PT-LAPTOP-NM-1W
PT-LAPTOP-NM-1W-A
PT-LAPTOP-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-WPC300N module provides one 2.4GHz wireless interface suitable for connection to wireless networks. The module supports protocols that use Ethernet for LAN access.

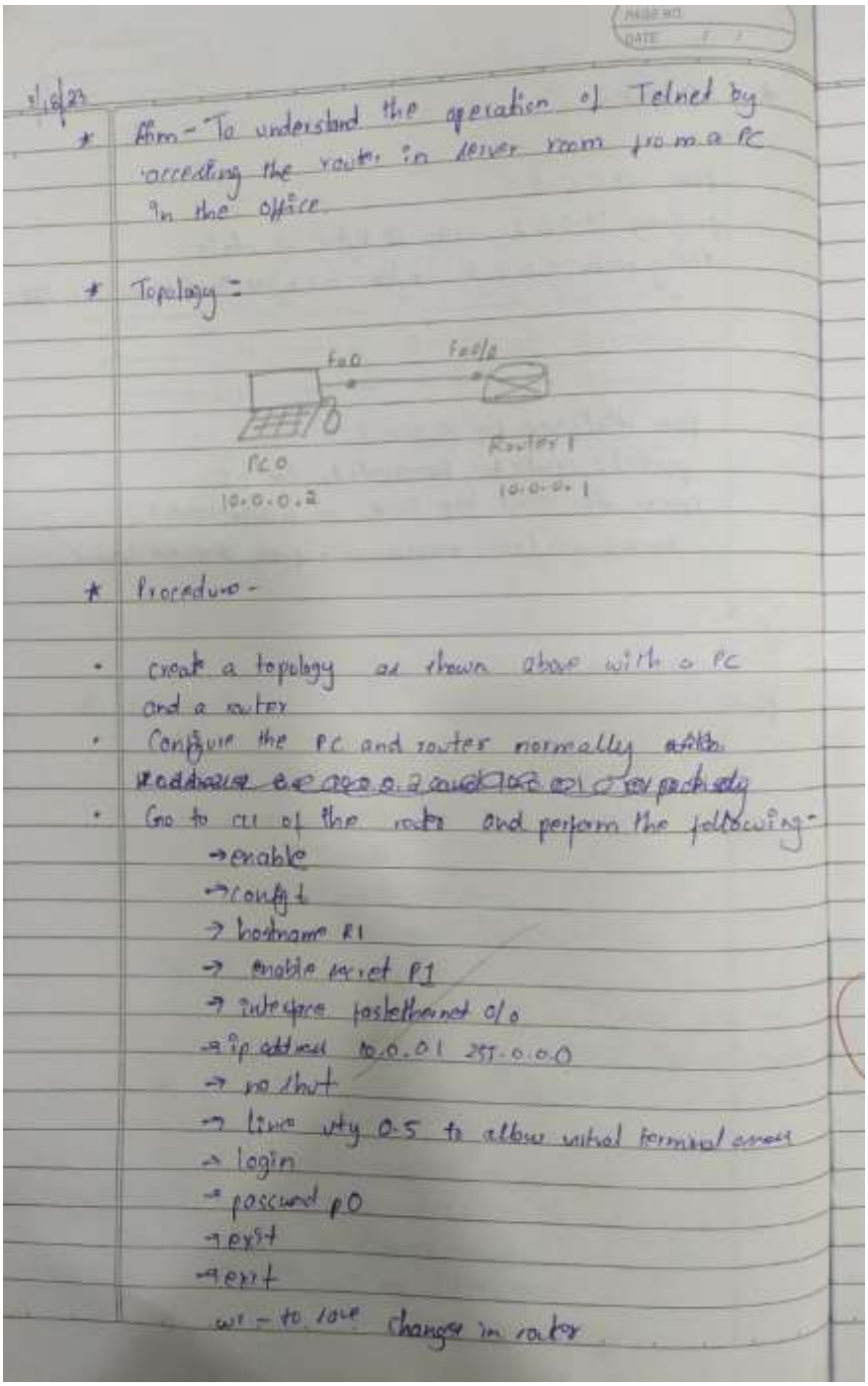
OUTPUT:


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 10.0.0.3:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>ping 10.0.0.3
Pinging 10.0.0.3 with 32 bytes of data:
Reply from 10.0.0.3: bytes=32 time=21ms TTL=128
Reply from 10.0.0.3: bytes=32 time=7ms TTL=128
Reply from 10.0.0.3: bytes=32 time=9ms TTL=128
Reply from 10.0.0.3: bytes=32 time=10ms TTL=128
Ping statistics for 10.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 7ms, Maximum = 21ms, Average = 11ms
PC>

WEEK 12

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

OBSERVATION:



1/8/23

* Aim - To understand the operation of Telnet by accessing the router in server room from a PC in the office

* Topology :

Fa0 Fa0/0

PC0 Router 1

10.0.0.2 10.0.0.1

* Procedure -

- create a topology as shown above with a PC and a router
- Configure the PC and router normally with ~~address 10.0.0.2 and 10.0.0.1 respectively~~
- Go to CLI of the router and perform the following -
  - → 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 0-5 to allow virtual terminal access
  - → login
  - → password p0
  - → exit
  - → exit
  - write to save changes in router

★ Commands in PC -

ping 10.0.0.1

Ping results seen -

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 = 0ms TTL = 255
——— " ——— " ———
——— " ——— " ———
——— " ——— " ———

ping statistics for 10.0.0.1
packets : sent = 4, Received = 4, loss = 0 [0% loss],
Approximate round trip in mili-seconds.
Minimum = 0ms, Maximum = 0ms, Average = 0ms.

PC > telnet 10.0.0.1
trying 10.0.0.1 ..... open.

user access verification.

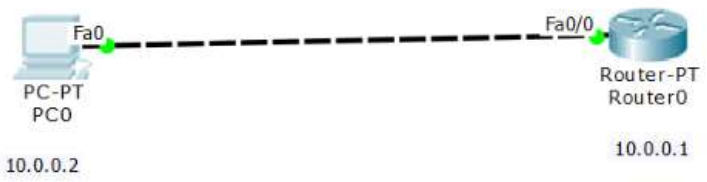
10/10

password : 10
r1 > enable
password : P1
r1# show ip route

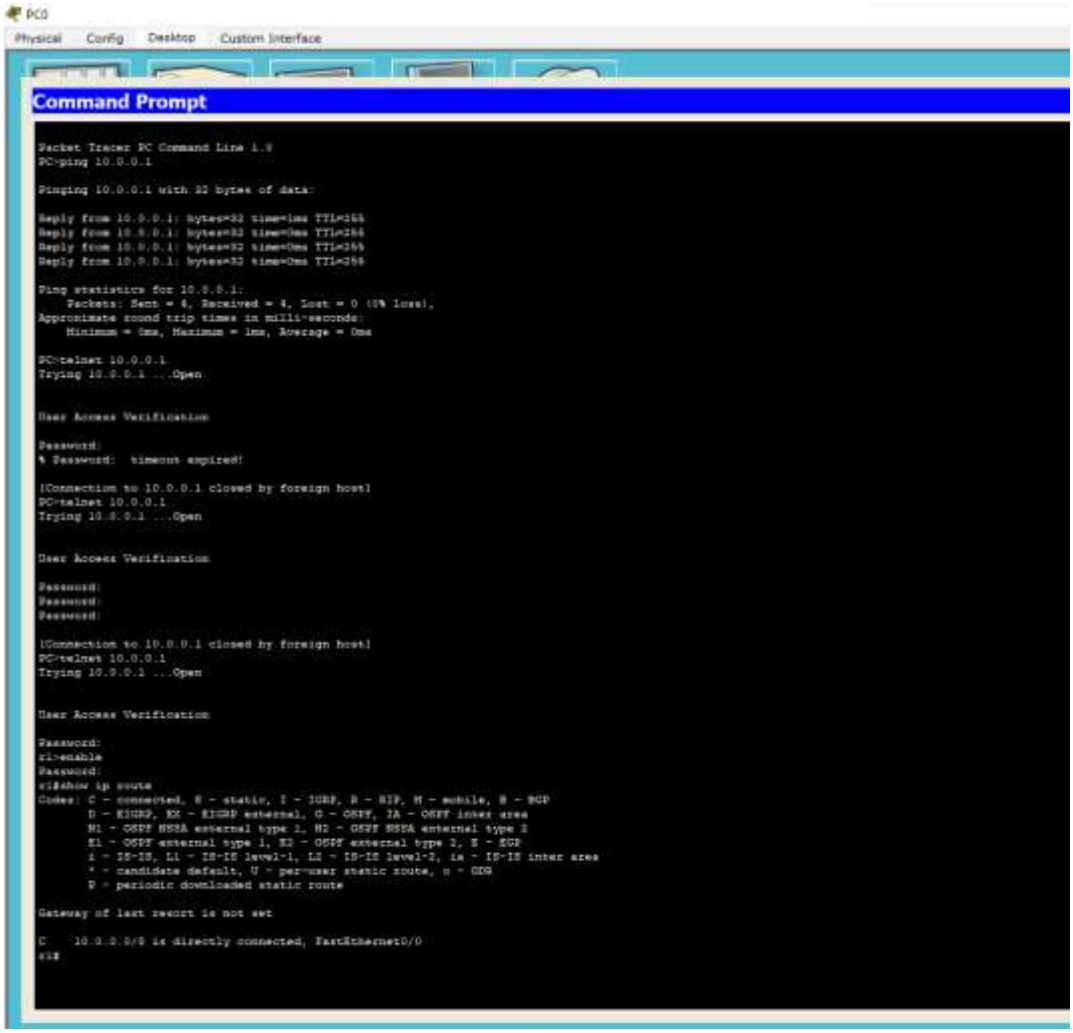
18/8

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

TOPOLOGY:



OUTPUT:



WEEK 13

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

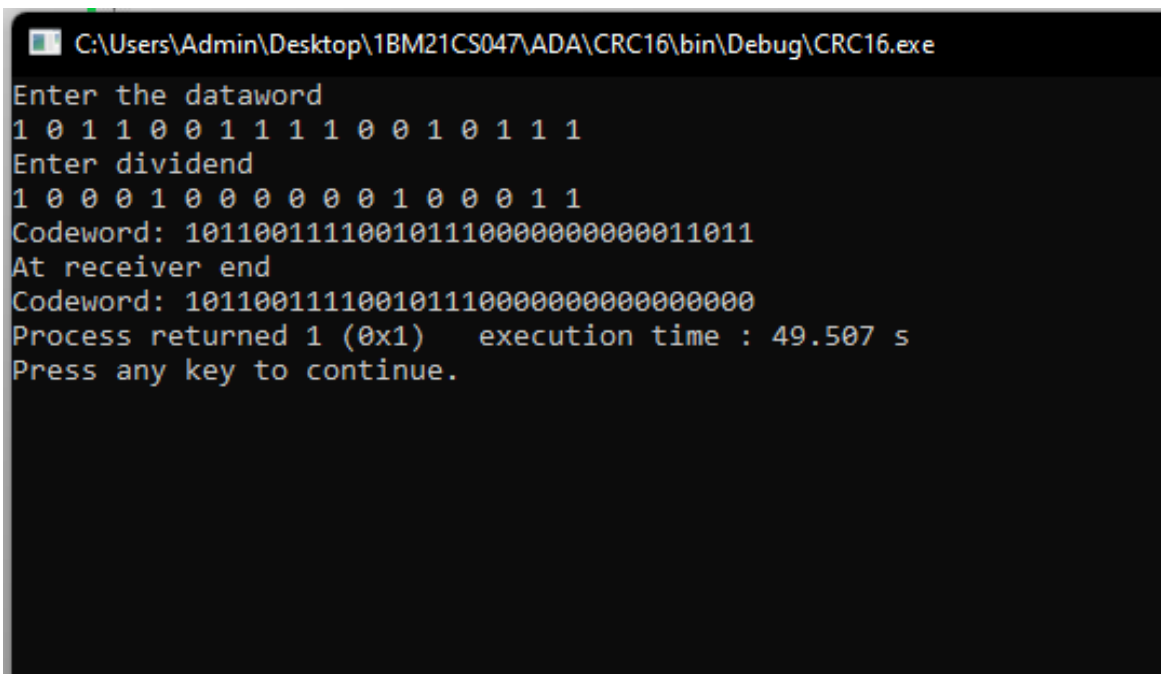
CODE:

```
#include<stdio.h>
int arr[17];

void xor(int x[], int y[])
{
    int k=0;
    for(int i=1;i<16;i++)
    {
        if(x[i]==y[i])
            arr[k++]=0;
        else
            arr[i]=1;
    }
}

void main()
{
    int dd[17],div[33],ze[17],i,k;

    printf("Enter the dataword \n");
    for(i=0;i<17;i++)
        scanf("%d",&div[i]);

    for(i=i;i<33;i++)
        div[i]=0;

    for(i=0;i<17;i++)
        ze[i]=0;
    printf("Enter dividend \n");
```

```
for(i=0;i<17;i++)
    scanf("%d",&dd[i]);

i=0;
k=0;
    for(i=i;i<17;i++)
        arr[k++]=div[i];
while(i<33)
{
    if(arr[0]==0)
        xor(arr,ze);
    else
        xor(arr,dd);

    arr[16]=div[i++];

}
k=0;
for(i=17;i<33;i++)
    div[i]=arr[k++];
printf("Codeword: ");
    for(i=0;i<33;i++)
        printf("%d",div[i]);

for(i=0;i<17;i++)
    arr[i]=0;

printf("\nAt receiver end \n");

 k=0;
    for(i=i;i<17;i++)
        arr[k++]=div[i];
while(i<33)
{
    if(arr[0]==0)
```

```
            xor(arr,ze);
        else
            xor(arr,dd);

        arr[16]=div[i++];

    }
    k=0;
    for(i=17;i<33;i++)
        div[i]=arr[k++];

    printf("Codeword: ");
        for(i=0;i<33;i++)
            printf("%d",div[i]);
}
```

OUTPUT:



OBSERVATION:

① 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 crc (int n)
{
int i,j;
for (i=0; i<n; i++)
    temp[i] = m[i];
for (i=0; i<n-16; i++) {
    if [r[0] == '1') {
    q[i] = '1';
      calcrem(); }
    else
        q[i] = '0';
        shift();
  }
  r[10] = m[17+i];
  r[17] = '10';
  for (j=0; j<=17; j++)
      temp[j] = r[j];
  }
  q [n-16] = '10';
  }
```

DATE / /

```
void calram()
  {
   int i,j;
    for(i=1; i<=10; 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 zatteant(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] = '10';
   }

   void main()
   {
    int n,i=0;
    char ch, flag=0;
    printf("enter frame bits");
     while ((ch = getch(stdin)) != '\n')
       m[i++] = ch;
        n = i;
      for(i=0; i<16; i++)
         m[n++] = '0';
        m[n] = '10';
```

```
printf("message after appending 16 zeros %s", m);
for(i=0; i<=16; i++)
    g[i] = '0';
g[0] = g[4] = g[11] = g[16] = '1';
    g[17] = '10';
printf("generator : %s\n", g);
    crc(n);
printf("quotient: %s", q);
    calfterans(n);
printf("transmitted frame : %s", m)
printf("enter transmitte frame :");
scanf("%s", m);
printf("crc checker\n");
crc(n);
printf("last remainder : %s", r);
for(i=0; i<16; i++)
    if (r[i] != '0')
    flag = '1';
    else
    continue;
if (flag == '1')
printf("error");
else
printf("frame all correct");
}
```

PAGE NO.
DATE / /

O/p

enter frame bits : 1011
Message after appending 16 zero
1011 0000 0000 0000 0000
generator : 10001000000100001
quotient : 1011
transmitted : 1011 1011 0001 0110 1011
enter - transmitted frame
1011 1011 0001 0110 1011
Last remainder 0000 0000 0000 0000
Received frame is correct

10/10

7/9/23

WEEK 14

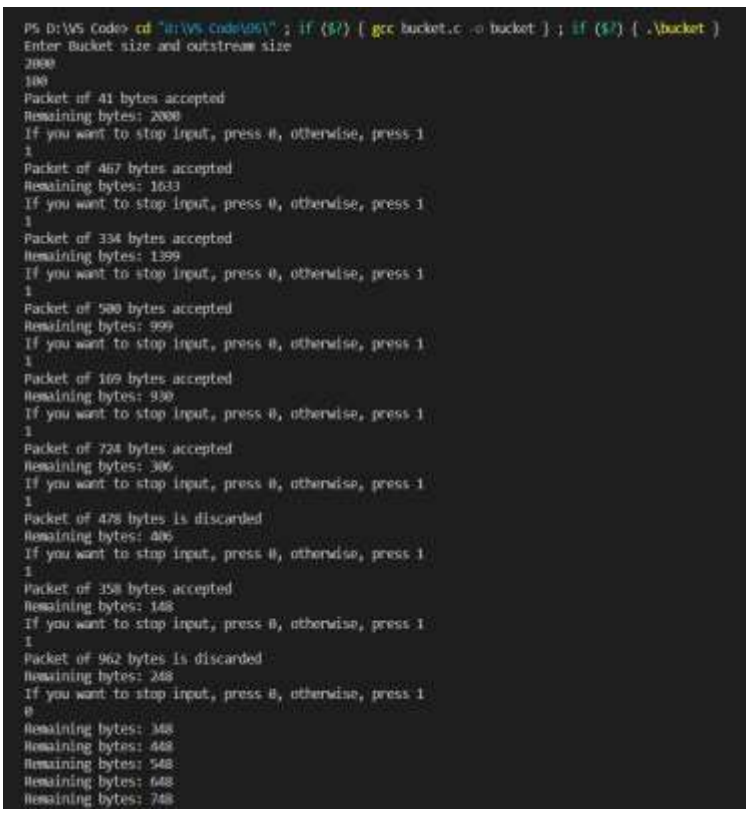
Write a program for congestion control using Leaky bucket algorithm.

CODE:

```
#include <stdio.h>
#include <stdlib.h> // Include this for the rand() function
int main()
{
    int buckets, outlets, k = 1, num, remaining;
    printf("Enter Bucket size and outstream size\n");
    scanf("%d %d", &buckets, &outlets);
    remaining = buckets;
    while (k)
    {
        num = rand() % 1000; // Generate a random number between 0 and 999
        if (num < remaining)
        {
            remaining = remaining - num;
            printf("Packet of %d bytes accepted\n", num); // Added missing variable
        }
        else
        {
            printf("Packet of %d bytes is discarded\n", num);
        }
        if (buckets - remaining > outlets)
        {
            remaining += outlets; // Fixed the calculation
        }
        else
            remaining = buckets;
        printf("Remaining bytes: %d \n", remaining);
        printf("If you want to stop input, press 0, otherwise, press 1\n");
        scanf("%d", &k);
```

```
    }
    while (remaining < buckets) // Fixed the condition
    {
        if (buckets - remaining > outlets)
        {
            remaining += outlets; // Fixed the calculation
        }
        else
            remaining = buckets;
        printf("Remaining bytes: %d \n", remaining);
    }
    return 0; // Added a return statement to indicate successful completion
}
```

OUTPUT:



```
PS D:\VS Code> cd "d:\VS Code\DS\" ; 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>
```

OBSERVATION:

Q WAP for congestion control using leaky bucket algorithm -

```
#include <stdio.h>
void main
{
int incoming, outgoing, bucketsize, n, store=0;
printf("enter bucket size, outgoing rate and
        no of i/p");
scanf("%d %d", &bucket_size, &outgoing, &n);
while (n!=0)
{
printf("enter incoming packet");
scanf("%d", &incoming);
printf("Incoming packet size %d\n", incoming);
if (incoming <= (bucket_size))
  {
  store += incoming;
printf("bucket buffer size %d out of %d", store,
    bucket_size);
    else {
printf("dropped %d no of packet\n",
        incoming-(bucket_size),
printf("bucket buffer size %d out of %d\n",
  store, bucket_size),
    store = bucket_size(),
  }
  store = store - outgoing;
printf("After outgoing %d packets left out %d in
    buffer, store bucket_size),
      n--;
  }
}
```

O/P -

enter bucket size, outgoing rate and no of I/P
20 10 2

Enter the incoming packet size = 30
Incoming packet size = 30
dropped 10 no. of packets
Bucket buffer size 0 out of 20
After outgoing 10 packet left out 20 in buffer

enter the incoming packet size = 10
Incoming packet size = 10
Bucket buffer size 0 out 20
After outgoing 10 packets left out 20 in buffer.

5 case

9/10
19/23

WEEK 15

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.

CODE:

ClientTCP.py

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

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 receive”)
connectionSocket, addr = serverSocket.accept()
sentence = connectionSocket.recv(1024).decode()
file=open(sentence,”r”)
l=file.read(1024)
connectionSocket.send(l.encode())
```

```
print ("\nSent contents of " + sentence)
file.close()
connectionSocket.close()
```

OUTPUT:

OBSERVATION:

Q) 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.

Server

1 A server has bind() method which binds to specific IP and port so that it can listen to incoming request on that IP port and port.

2 A server has a listen() method which puts the server into listening mode. This allows server to the listen to incoming connections.

3 Server has accept() and close() method.
accept() - initiates a connection with client
close() - closes the connection with client.

Step 1: Open idle, in that in file, open new file and write the following code and same as server.py
ser_TCP.py

```
from socket import *
server Name = "127.0.0.1"
server Port = 12000
server socket = socket (AF_INET,
                        SOCKET_STREAM)
server socket.bind ((server Name, server Port))
server socket listen()
while 1:
print('server is ready to receive');
connection socket, addr = server socket accept()
sentence = connection socket.recv (02, decode())
```

PAGE NO.
DATE / /

```
file = open (sentence, "r")
  l = file read (1024)
connection socket. send (l. encode())
print("In sent contents of + 1 entence)
      file. close()
   connection socket. close()
```

Step 2: Run the file server.py
o/p ⇒ The server is ready to receive
This shows that server is working

Client.

Step 1: Make a socket object
Step 2: establish a connection with server and lastly we will receive data from the server and close the function.
Step 3: Open idle and open new file and write "client.py".

```
from socket impact *
server Name = "127.0.0.1"
server port = 12000
client socket = socket( AF_INET, SOCK_STREAM)
client socket. connect ((server Name, server PORT))
sentence = input ("In enter file name: ")
client socket. send ( sentence. encode())
file contents = client socket. recv (1024). decode()
print ("I from server: In)"
print ("file contents)
    client socket. close()
```

Run the file client Py
o/p => enter file name. server. Py

WEEK 16

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.

CODE:

ClientUDP.py

```
from socket import *
serverName = “127.0.0.1”
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input(“\nEnter file name: “)
clientSocket.sendto(bytes(sentence,”utf-8”),(serverName, serverPort))
filecontents,serverAddress = clientSocket.recvfrom(2048)
print (“\nReply from Server:\n”)
print (filecontents.decode(“utf-8”))
# for i in filecontents:
# print(str(i), end = “ ”)
clientSocket.close()
clientSocket.close()
```

ServerUDP.py

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

```
con=file.read(2048)
serverSocket.sendto(bytes(con,”utf-8”),clientAddress)
print (“\nSent contents of “, end = “ “)
print (sentence)
# for i in sentence:
# print (str(i), end = “ “)
file.close()
```

OUTPUT:

OBSERVATION:

10/10

server o/p -
The server is ready to receive sent the contents of server.py
The server is ready to receive.

Q) Using UDP sockets, write a client server program to make client sending the file name and the server to send back the contents of req file if present

Here, like in TCP/IP we create socket object and bind it to specified port and server will be continuously listening ~~[illegible]~~ when the client sends request it responds accordingly

```
server UDP.py
  from socket import *
  server port = 12000
server socket = socket(AF_INET, SOCK_DGRAM)
  server socket bind = (("127.0.0.1", server port))
  print("The server is ready to receive").
  while 1:
  sentence, client Address = server socket.
      recevefrom (2048)
  sentence = sentence. decode ("utf-8")
      file = open( sentence, "r")
      con = file. read (2048)
      server socket. sendto (bytes (con, "utf-8");
          client Address)
  print("\n sent contents of ", end = ")
      print(sentence)
```

PAGE NO.
DATE / /

client UDP.py

```
from socket import *
server name = "127.0.0.1"
server Port = 12000
client socket = socket (AF_INET, SOCK_DGRAM)
sentence = input ("\n enter file name:")
client socket.send to (bytes(sentence, "utf-8"),
                        (server Name, server port))
file content, server Address = client socket.
                                recvfrom (2048)
print ("\n Reply from server:\n")
print (file content.decode("utf-8"))
```

o/p (~~element~~ client)

enter the file name: server UDP.py.

o/p (server)

The server is ready to receive
sent contents of server UDP.py.

10/10

2/9/23