`

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT

on

COMPUTER NETWORKS

Submitted by

Gursimranjeet Singh(1BM22CS104)

in partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

(Autonomous Institution under VTU)

BENGALURU-560019

Sep 2024-Jan 2025

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**COMPUTER NETWORKS**" carried out by **Gursimranjeet Singh(1BM22CS104),** who is bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2024-25. The Lab report has been approved as it satisfies the academic requirements in respect of **Computer Networks Lab - (23CS5PCCON)** work prescribed for the said degree.

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

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

INDEX

CYCLE 1

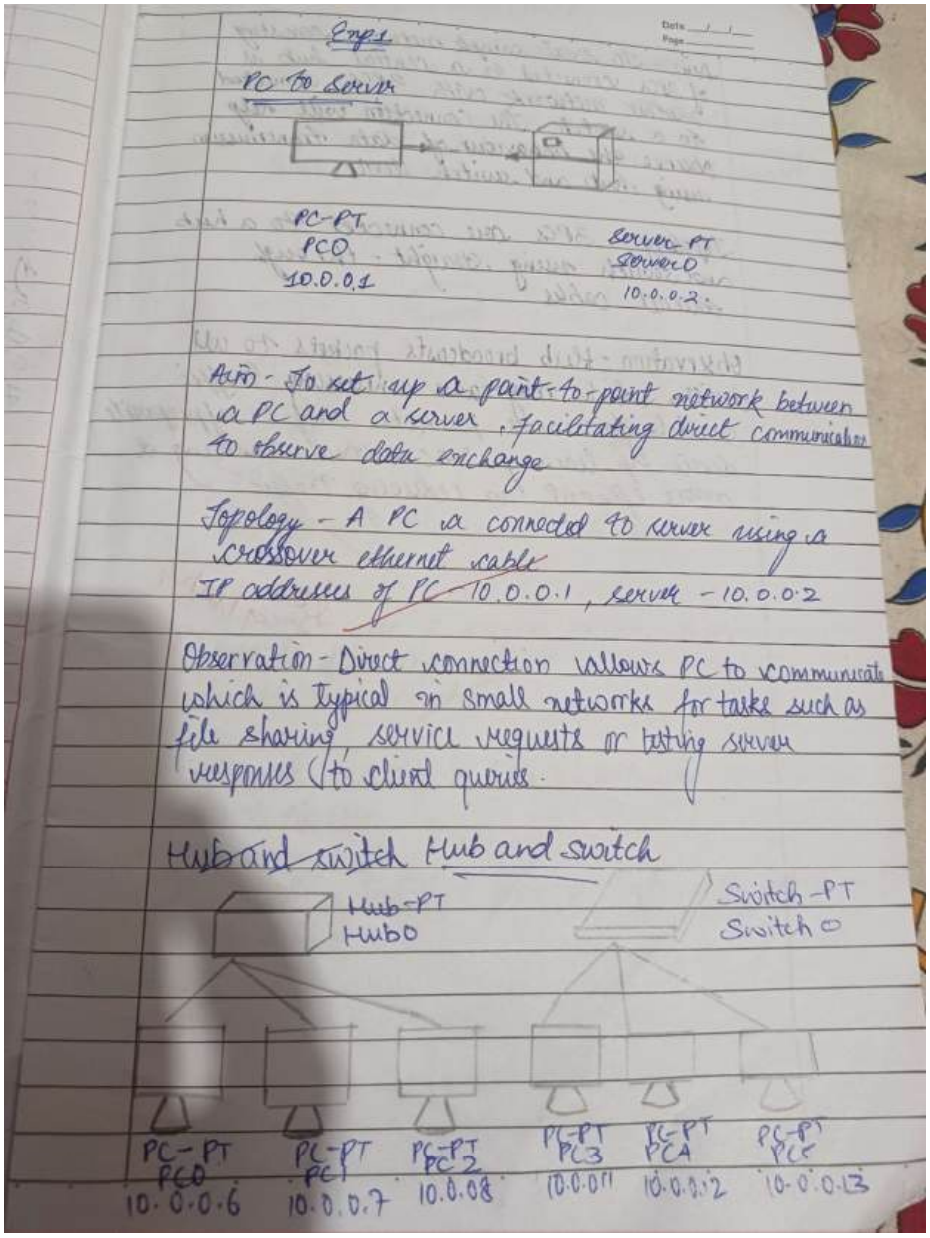
INDEX

CYCLE 2

CYCLE-1

PROGRAM1: Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.

OBSERVATION



Exp1

PC to Server

PC-PT
PC0
10.0.0.1

Server-PT
Server0
10.0.0.2.

Aim - To set up a point-to-point network between a PC and a server, facilitating direct communication to observe data exchange

Topology - A PC is connected to server using a crossover ethernet cable
IP addresses of PC - 10.0.0.1, server - 10.0.0.2

Observation - Direct connection allows PC to communicate which is typical in small networks for tasks such as file sharing, service requests or letting server responses (to client queries.

~~Hub and switch~~ Hub and switch

Hub-PT
Hub0

Switch-PT
Switch0

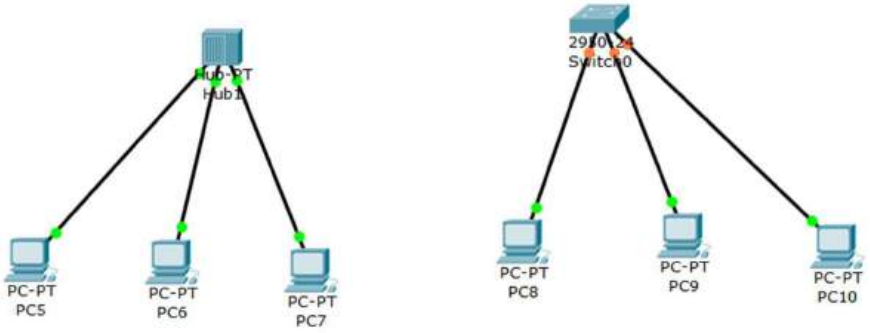
PC-PT PC0 10.0.0.6
PC-PT PC1 10.0.0.7
PC-PT PC2 10.0.0.8
PC-PT PC3 10.0.0.11
PC-PT PC4 10.0.0.12
PC-PT PC5 10.0.0.13

Aim - To create sample network consisting of 3PCs connected to a central hub is another network with 3PCs connected to a switch. The connection will help observe the behaviour of data transmission using hub and switch device

Topology - 3PCs are connected to a hub and switch using straight-through ethernet cables

Observation - Hub broadcasts packets to all devices which may cause unnecessary traffic switch forwards packets only to appropriate device by learning MAC address, making it more efficient in reducing traffic

TOPOLOGY:

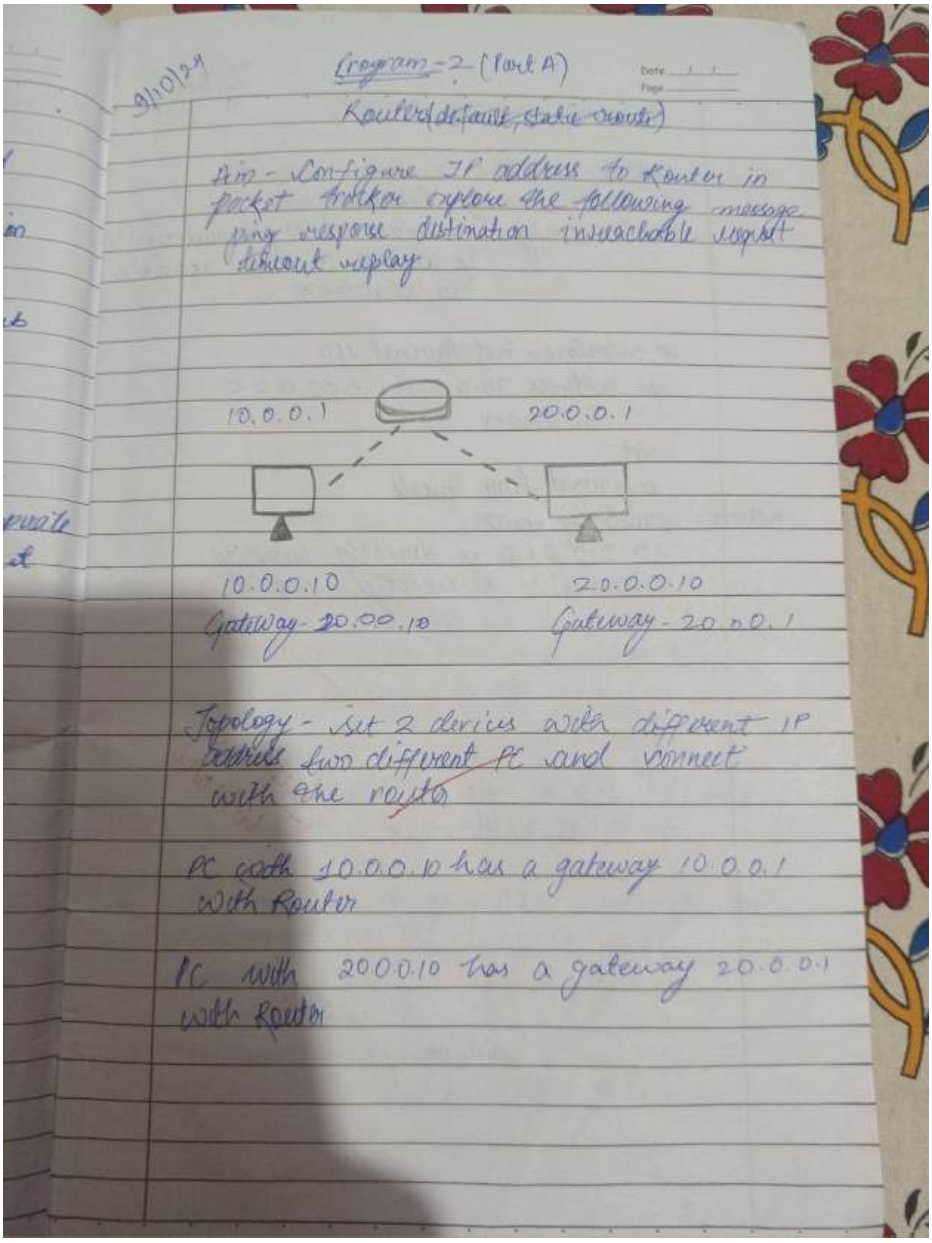


OUTPUT:

Pcs are connected

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

OBSERVATION:



9/10/24

Program - 2 (Part A)

Router (default, static route)

Aim - Configure IP address to Router in packet tracer explore the following message ping response destination unreachable request timeout reply

10.0.0.1 20.0.0.1

10.0.0.10 20.0.0.10

Gateway - 20.00.10 Gateway - 20.0.0.1

Topology - set 2 devices with different IP address two different PC and connect with one router

PC with 10.0.0.10 has a gateway 10.0.0.1 with Router

PC with 20.0.0.10 has a gateway 20.0.0.1 with Router

Procedure

router - enable
router # - config terminal
router config - interface fast ethernet 0/0
ip address 10.0.0.1 255.0.0.
no shutdown

interface fast ethernet 1/0
ip address 20.0.0.1 255.0.0.0
no shutdown
exit
Configured from console

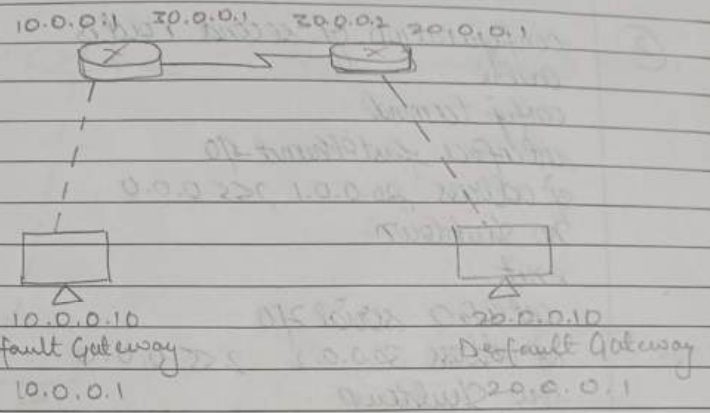
Result: show ip route
10.0.0.0/8 is directly connected
20.0.0.0/8 is directly connected

16/10/24

16/10/21

~~LAB-3~~ PART-B

Configure default route, static route to the Router.



Topology - Two PCs are connected of two routers respectively by means of copper cross. The routers are connected by serial coring (DCE)

Procedure

① Take two routers and connect the two systems to two router respectively.

② In first router open CLI and type below command

- Enable
- config terminal
- interface fastEthernet 0/0
- ip address 10.0.0.1 255.0.0.0
- no shutdown
- exit

interface Serial 2/0
ip address 30.0.0.1 255.0.0.0
no shutdown
exit

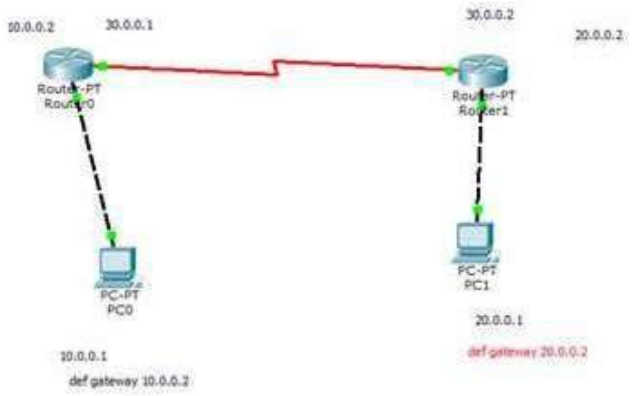
(2) configuration of second router
enable
config terminal
interface fastEthernet 1/0
ip address 20.0.0.1 255.0.0.0
no shutdown
exit
interface serial 2/0
ip address 30.0.0.2 255.0.0.0
no shutdown
exit

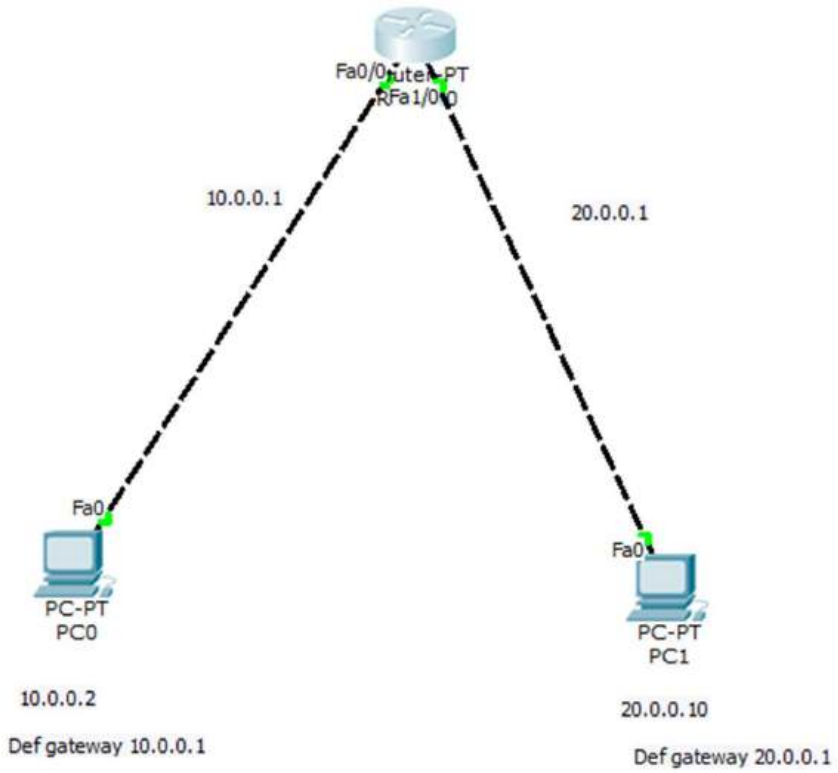
ping
command prompt
ping 20.0.0.1
reply from 20.0.0.10: Destination host unreachable

configuring static route
ip route 10.0.0.0 255.0.0.0 30.0.0.1

ping
command prompt
ping 20.0.0.1
Reply from 20.0.0.10 byte=32 time=5ms TTL=

TOPOLOGY:


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


Fa0/0
Router-PT
RFa1/0/0
10.0.0.1
20.0.0.1
Fa0
Fa0
PC-PT
PC0
PC-PT
PC1
10.0.0.2
20.0.0.10
Def gateway 10.0.0.1
Def gateway 20.0.0.1

OUTPUT:

PC0

Physical | Config | Desktop | Custom Interface

Command Prompt

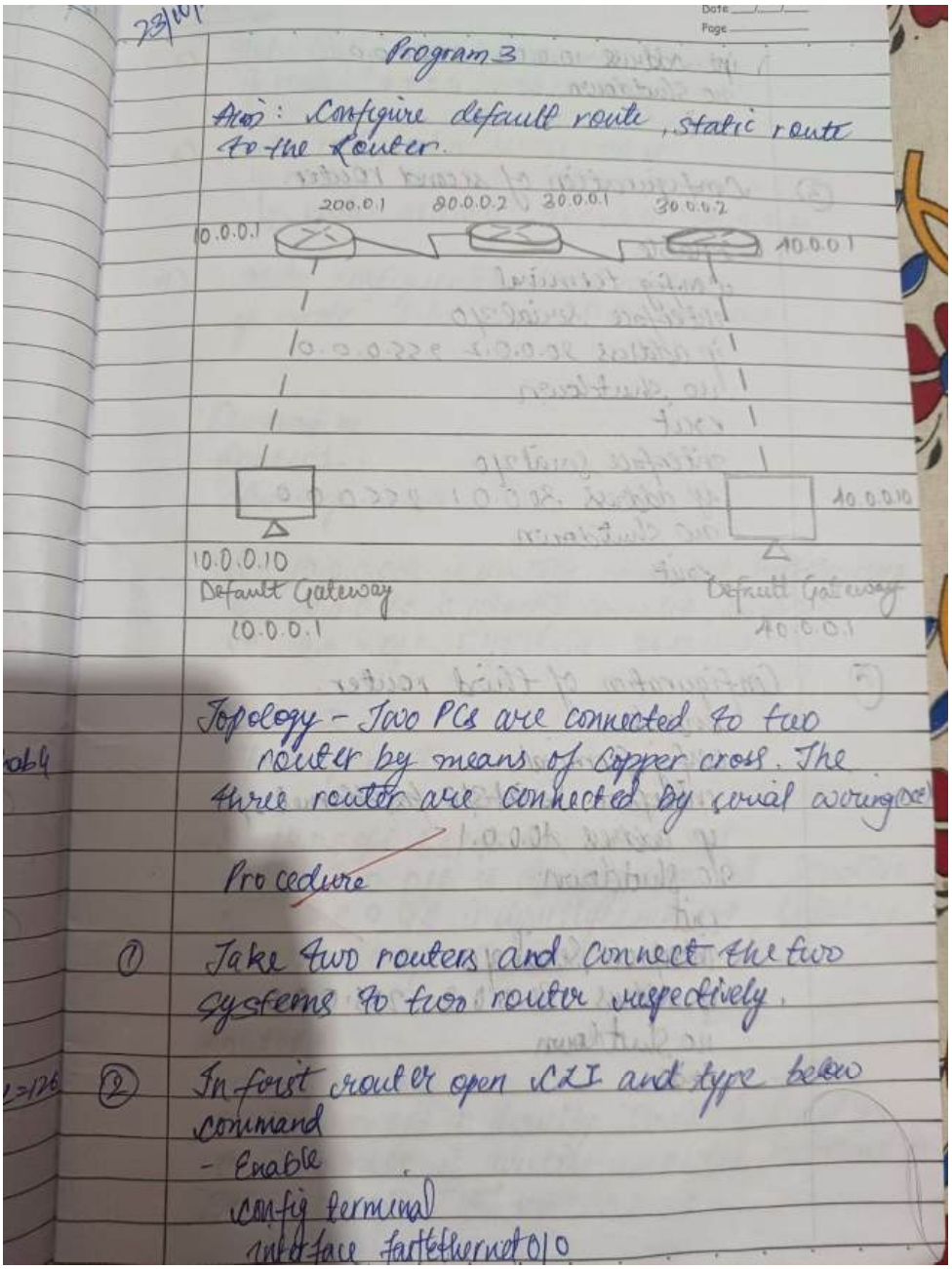
```
Pinging 20.0.0.10 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.10:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>ping 20.0.0.10

Pinging 20.0.0.10 with 32 bytes of data:

Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127
Reply from 20.0.0.10: bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

PC>
```

```
PC>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

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

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

PC>ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data:

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

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

```
PC>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.
Reply from 10.0.0.2: Destination host unreachable.

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

PC>ping 30.0.0.2

Pinging 30.0.0.2 with 32 bytes of data:

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

Ping statistics for 30.0.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

PC>
```

PROGRAM3: Configure default route, static route to the Router

OBSERVATION:



Program 3

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

200.0.1 80.0.0.2 30.0.0.1 30.0.0.2

10.0.0.1 10.0.0.1

10.0.0.10 10.0.0.10

Default Gateway 10.0.0.1

Default Gateway 10.0.0.1

Topology - Two PCs are connected to two router by means of copper cross. The three router are connected by serial wiring.

Procedure

1. Take two routers and connect the two systems to two router respectively.

2. In first router open CLI and type below command
   - Enable
   - config terminal
   - interface fastethernet 0/0

5) Static Configuration first router
ip route 0.0.0.0 0.0.0.0 20.0.0.2

6) Static configuration second router
ip route 10.0.0.0 255.0.0.0 20.0.0.1
ip route 40.0.0.0 255.0.0.0 30.0.0.2

7) Static configuration third router
ip route 0.0.0.0 0.0.0.0 30.0.0.1

Observation
Router1.
show ip route.

C 10.0.0.0/8 is directly connected, FastEthernet0/0
C 20.0.0.0/8 is directly connected, Serial2/0
S* 0.0.0.0/0 [1/0] via 20.0.0.2

Router 2
show ip route

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

Router 3
show ip route.
C 30.0.0.0/8 is directly connected, Serial2/0
C 40.0.0.0/8 is directly connected, FastEthernet1/0
S* 0.0.0.0/0 [1/0] via 30.0.0.1

ipt address 10.0.0.1 255.0.0.0
no shutdown
exit

(3) Configuration of second router.

enable
config terminal
interface Serial 3/0
ip address 20.0.0.2 255.0.0.0
no shutdown
exit
interface Serial 2/0
ip address 30.0.0.1 255.0.0.0
no shutdown
exit

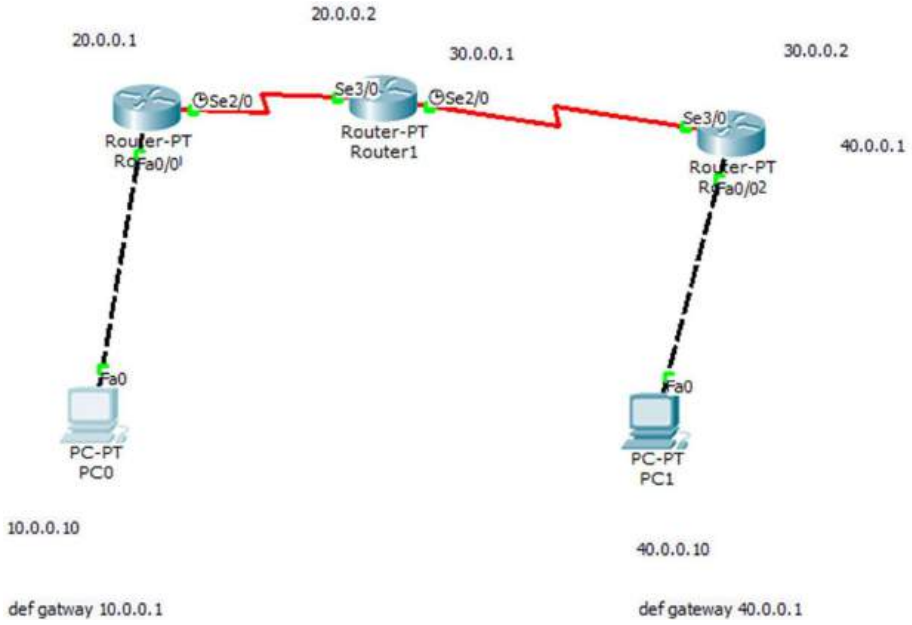
(4) Configuration of third router.

enable
config terminal
interface Serial 3/0 fastEthernet 1/0
ip address 40.0.0.1
no shutdown
exit
interface Serial 2/0
ip address 30.0.0.2 255.0.0.0
no shutdown
exit.

Ping 40 10.0.0.10
Ping statistics for 10.0.0.10
Packets: Sent=4, Received=4, Lost=0

23/10/24

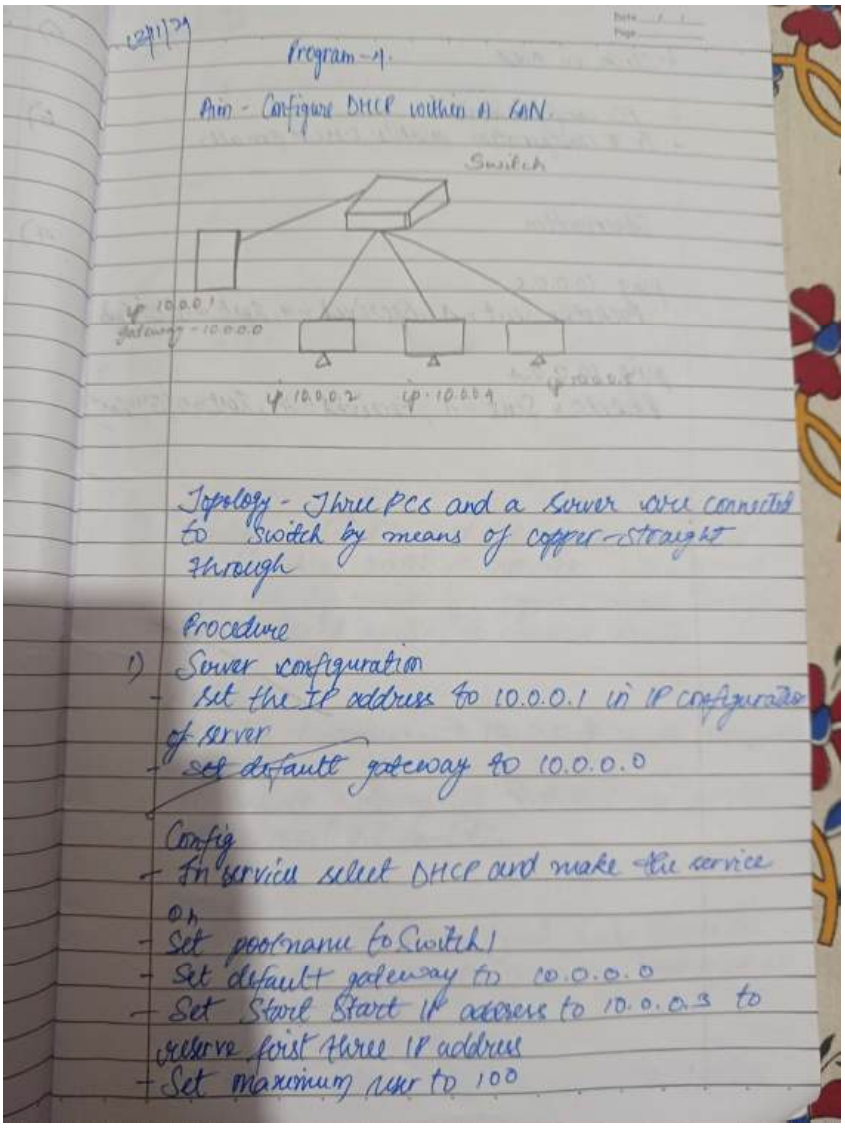
TOPOLOGY:


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

OUTPUT:

```
Packet Tracer PC Command Line 1.0
PC>ping 30.0.0.2

Pinging 30.0.0.2 with 32 bytes of data:

Reply from 30.0.0.2: bytes=32 time=6ms TTL=253
Reply from 30.0.0.2: bytes=32 time=7ms TTL=253
Reply from 30.0.0.2: bytes=32 time=8ms TTL=253
Reply from 30.0.0.2: bytes=32 time=7ms TTL=253

Ping statistics for 30.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 6ms, Maximum = 8ms, Average = 7ms

PC>ping 20.0.0.2

Pinging 20.0.0.2 with 32 bytes of data:

Reply from 20.0.0.2: bytes=32 time=5ms TTL=254
Reply from 20.0.0.2: bytes=32 time=3ms TTL=254
Reply from 20.0.0.2: bytes=32 time=3ms TTL=254
Reply from 20.0.0.2: bytes=32 time=3ms TTL=254

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 5ms, Average = 3ms

PC>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1: bytes=32 time=8ms TTL=253
Reply from 40.0.0.1: bytes=32 time=7ms TTL=253
Reply from 40.0.0.1: bytes=32 time=7ms TTL=253
Reply from 40.0.0.1: bytes=32 time=8ms TTL=253

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 7ms, Maximum = 8ms, Average = 7ms

PC>
```

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



Program - 4.

Aim - Configure DHCP within a LAN.

Switch

IP 10.0.0.1
Gateway - 10.0.0.0

IP 10.0.0.2 IP 10.0.0.4

Topology - Three PCs and a Server were connected to Switch by means of copper-straight through

Procedure

1) Server configuration
- set the IP address to 10.0.0.1 in IP configuration of server
- set default gateway to 10.0.0.0

Config
- In service select DHCP and make the service on
- Set pool name to Switch1
- Set default gateway to 10.0.0.0
- Set Start Start IP address to 10.0.0.3 to reserve first three IP address
- Set maximum user to 100

- Click on Add.

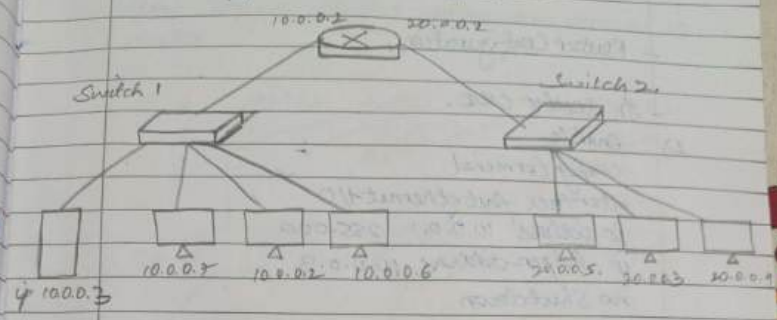
- PC configuration
- In IP configuration enable DHCP for all.

Observation

ping 10.0.0.5
Packets: Sent = 4, Received = 4, Lost = 0 (0%

ping 10.0.0.4
Packets: Sent = 4, Received = 4, Lost = 0 (0%

Aim - Configure DHCP outside LAN



Topology - Six PCs and a server are connected to respective switch of network 10.0.0.0 and 20.0.0.0 by means of copper straight through
- Connected each switch to common router

Procedure
- Set the IP address to 10.0.0.3 in IP configuration of server
- Set default gateway to 10.0.0.1 for switch1 and 20.0.0.1 for switch2

Config
- In services select DHCP and make service on
- Set default gateway to 10.0.0.1 and start IP address to 10.0.0.3.
Add pool
- Set default gateway to 20.0.0.1 for another

pool and start address to 20.0.0.3

- Router Configuration.

- In Router CLI.

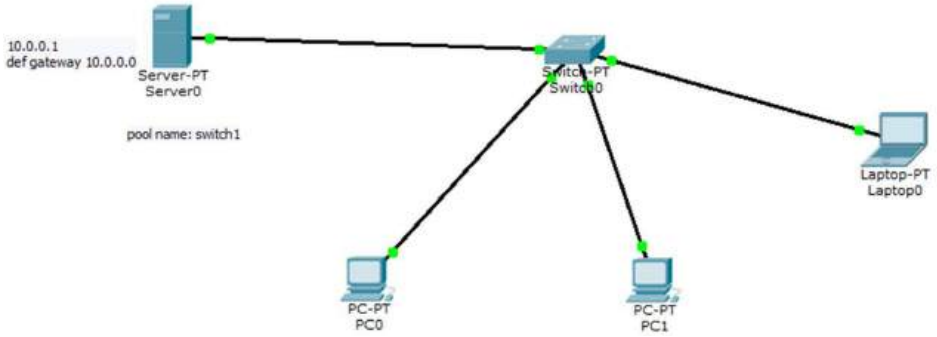
1) enable
config terminal
interface fastethernet 1/0
ip address 10.0.0.1 255.0.0.0
ip helper-address 10.0.0.3
no shutdown
exit

2) interface fastethernet 0/0
ip address 20.0.0.1 255.0.0.0
ip helper-address 10.0.0.3
no shutdown
exit
exit

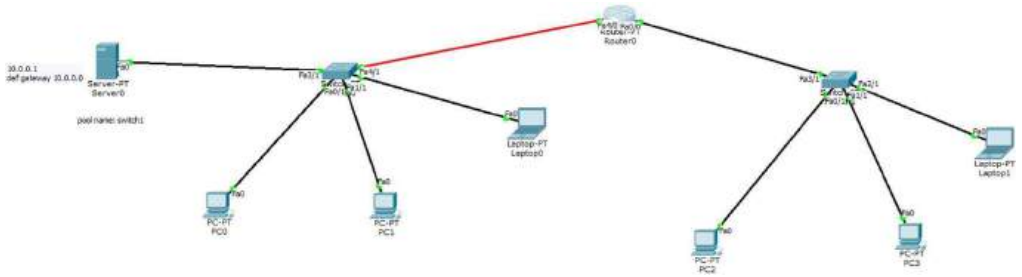
Observation
Show ip route
C 10.0.0.0/8 is directly connected, fastethernet
C 20.0.0.0/8 is directly connected, fastethernet

TOPOLOGY:

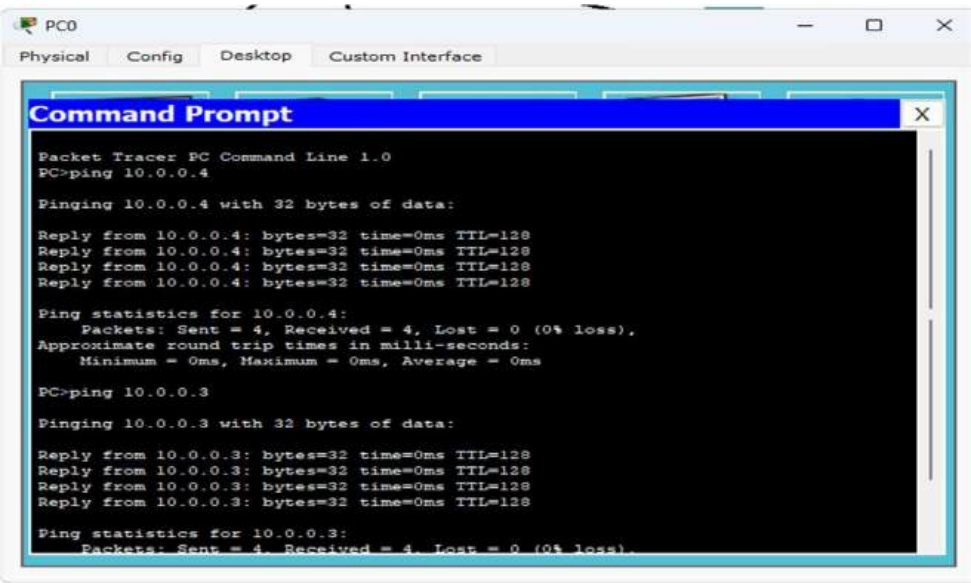
Within lan



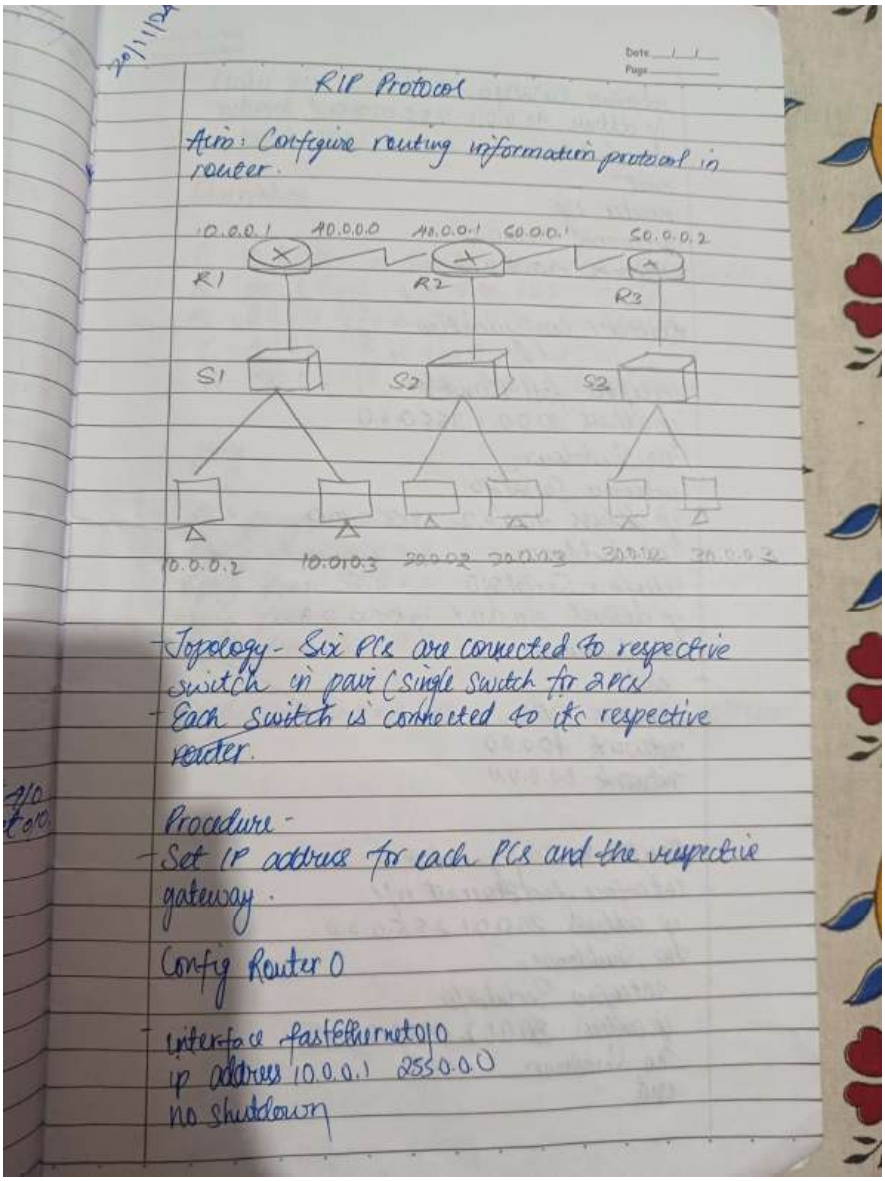
Outside lan



OUTPUT:



PROGRAM5: Configure RIP routing Protocol in Routers
OBSERVATION:



RIP Protocol

Aim: Configure routing information protocol in router.

10.0.0.1 40.0.0.0 40.0.0.1 60.0.0.1 60.0.0.2
R1 R2 R3
S1 S2 S3
10.0.0.2 10.0.0.3 20.0.0.2 20.0.0.3 30.0.0.2 30.0.0.3

- Topology - Six PCs are connected to respective switch in pair (single switch for 2 PCs)
- Each switch is connected to its respective router.

Procedure -
- Set IP address for each PCs and the respective gateway.

Config Router 0

- interface fastEthernet 0/0
ip address 10.0.0.1 255.0.0.0
no shutdown

interface Serial2/0
ip address 40.0.0.1 255.0.0.0
no shutdown.
exit
router rip
network 10.0.0.0
network 40.0.0.0

Router1 configuration

- interface fastEthernet 0/1
ip address 20.0.0.1 255.0.0.0
no shutdown.
- interface Serial2/0
ip address 40.0.0.2 255.0.0.0
no shutdown.
- interface Serial 3/0
ip address 50.0.0.1 255.0.0.0
no shutdown.
- exit
router rip
network 40.0.0.0
network 50.0.0.0

Router 2 configuration
- interface fastEthernet 0/1
ip address 30.0.0.1 255.0.0.0
no shutdown.
- interface Serial 2/0
ip address 50.0.0.2 255.0.0.0
no shutdown
exit

Date __/__/__
Page _____

router rip
network 30.0.0.0
network 50.0.0.0

Observation

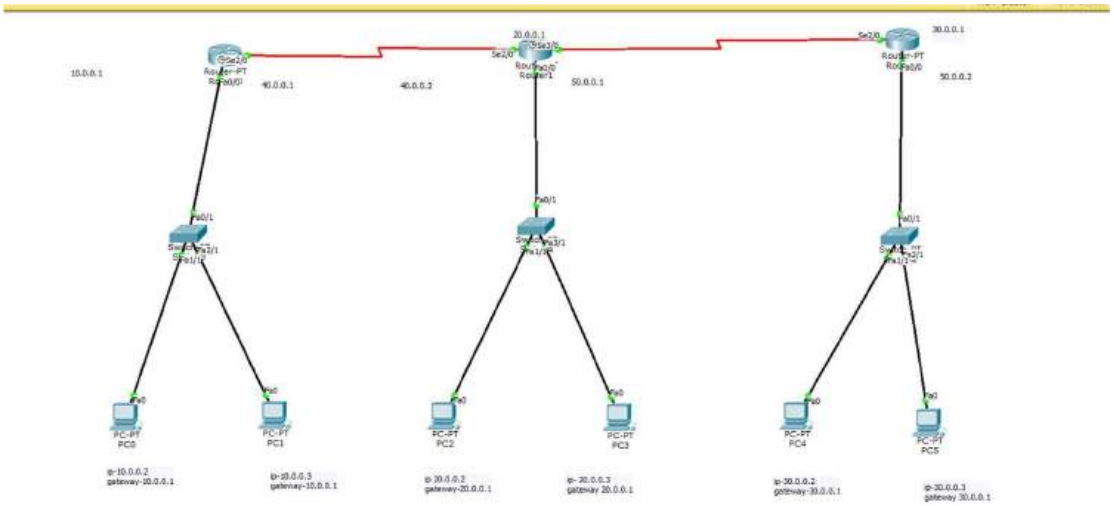
C 10.0.0.0/8 is directly connected, fastEthernet0/0
R 20.0.0.0/8 via 40.0.0.2, Serial2/0
R 30.0.0.0/8 via 40.0.0.2, Serial2/0
C 40.0.0.0/8 is directly connected.
R 50.0.0.0/8 via 40.0.0.2, Serial2/0

ping

PC> ping 30.0.0.3
Reply from 30.0.0.3
Reply from 30.0.0.3
Reply from 30.0.0.3
Reply from 30.0.0.3

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

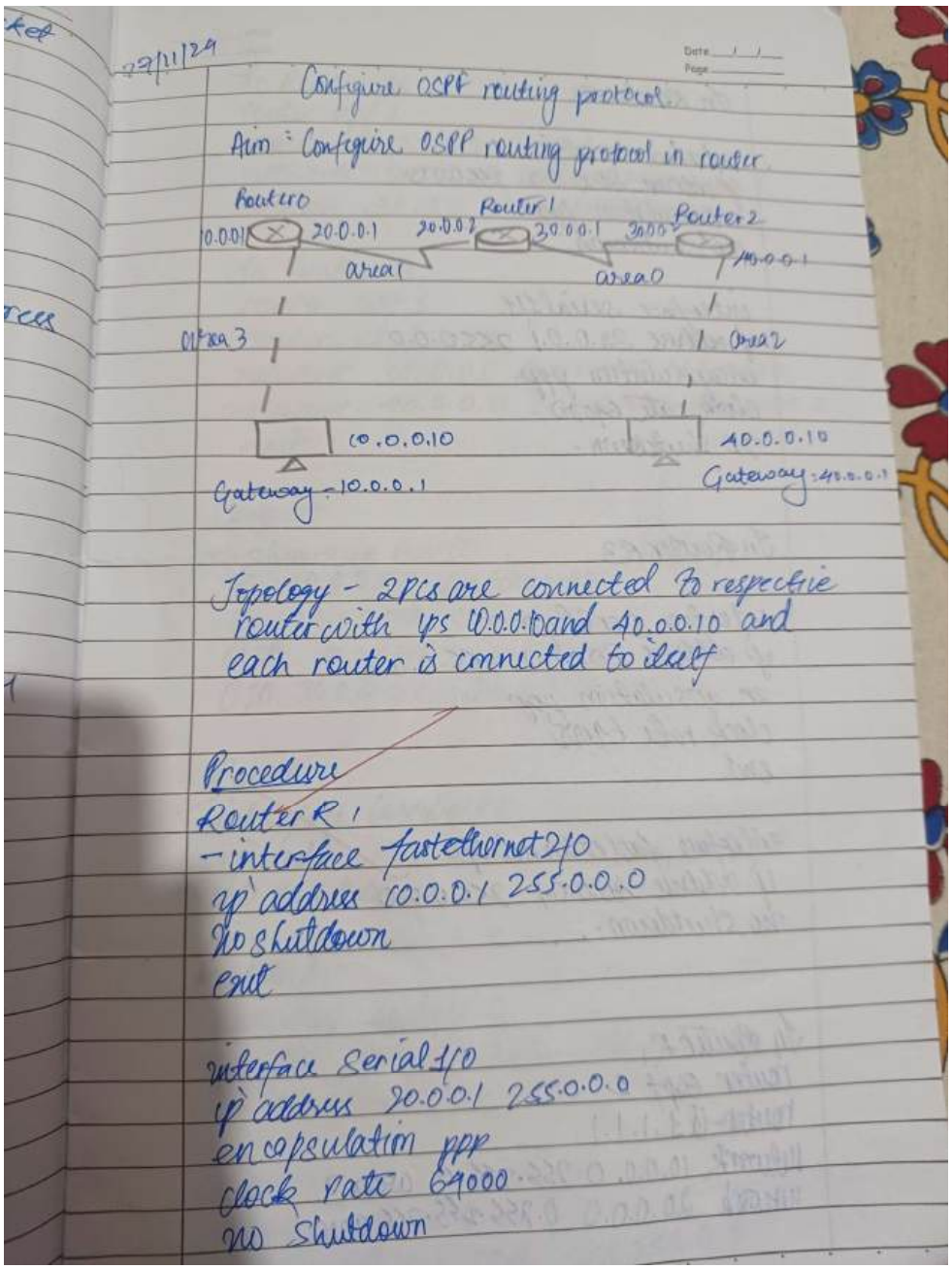
TOPOLOGY:



OUTPUT:

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

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

PC>ping 20.0.0.2

Pinging 20.0.0.2 with 32 bytes of data:

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

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

PC>
```

PROGRAM6: Configure OSPF routing protocol
OBSERVATION:



27/11/24

Configure OSPF routing protocol.

Aim : Configure OSPF routing protocol in router

Router0 — Router1 — Router2

10.0.0.1 — 20.0.0.1 — 20.0.0.2 — 30.0.0.1 — 30.0.0.2 — 40.0.0.1

area1 — area0

area3 — area2

10.0.0.10 — 40.0.0.10

Gateway - 10.0.0.1 — Gateway : 40.0.0.1

Topology - 2 PCs are connected to respective router with ips 10.0.0.10 and 40.0.0.10 and each router is connected to itself

Procedure

Router R1

- interface fastethernet 2/0
ip address 10.0.0.1 255.0.0.0
no shutdown
exit

interface serial 1/0
ip address 20.0.0.1 255.0.0.0
encapsulation ppp
clock rate 64000
no shutdown

In Router R2

```
interface serial1/0
ip address 20.0.0.2 255.0.0.0
encapsulation ppp
no shutdown.

interface serial1/1.
ip address 30.0.0.1 255.0.0.0
encapsulation ppp.
clock rate 64000
no shutdown.
```

In Router R3

```
interface serial1/0
ip address 30.0.0.2 255.0.0.0
encapsulation ppp
clock rate 64000
exit

interface fastethernet 2/0
ip address 40.0.0.1 255.0.0.0
no shutdown.
```

In Router R1

```
router ospf 1
router-id 1.1.1.1
network 10.0.0.0 255.255.0 area 3
network 20.0.0.0 0.255.255.255 area 1
```

In Router R2
router ospf 1
router-id 2.2.2.2
network 20.0.0.0 255.255.255 area 1
network 30.0.0 0.255.255.255 area 0

In Router R3
router ospf 1
router-id 3.3.3.3
network 30.0.0.0 0.255.255.255 area 0
network 40.0.0.0 0.255.255.255 area 2.
exit

Output
#show ip route
C 10.0.0.0/8 FastEthernet 2/0
C 20.0.0.0/8 Serial1/0
O IA 40.0.0.0/8 Serial1/0
O IA 30.0.0.0/8 Serial1/0

Router 1
interfac loopback 0
ip add 172.16.1.252 255.255.0.0
no shutdown

Router 2
interfac loopback 0
ip add 172.16.1.253 255.255.0.0
no shutdown

Router 3
interfac loopback 0
ip add 172.16.1.254 255.255.0.0

Observation

O IA 20.0.0.0/8 Serial1/0
C 40.0.0.0/8 FastEthernet2/0
C 30.0.0.0/8 Serial1/0

Router1

router ospf 1
area 1 virtual-link 2.2.2.2.
Loading Done

In Router2
virtual link not found
Loading Done

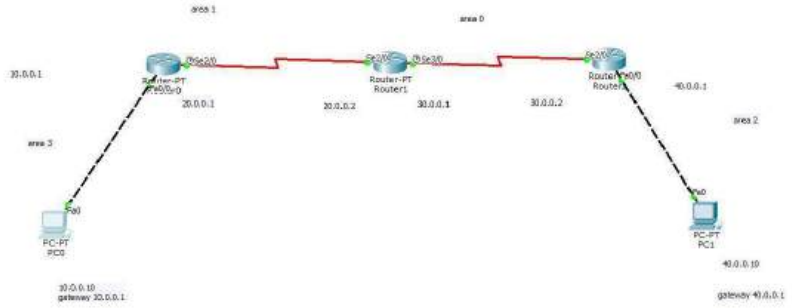
R2# area 1 virtual-link 1.1.1.1
exit

Observation

ping 40.0.0.10
Packets Sent=4, Received=4, Lost=0 (0% loss)

27/11/20

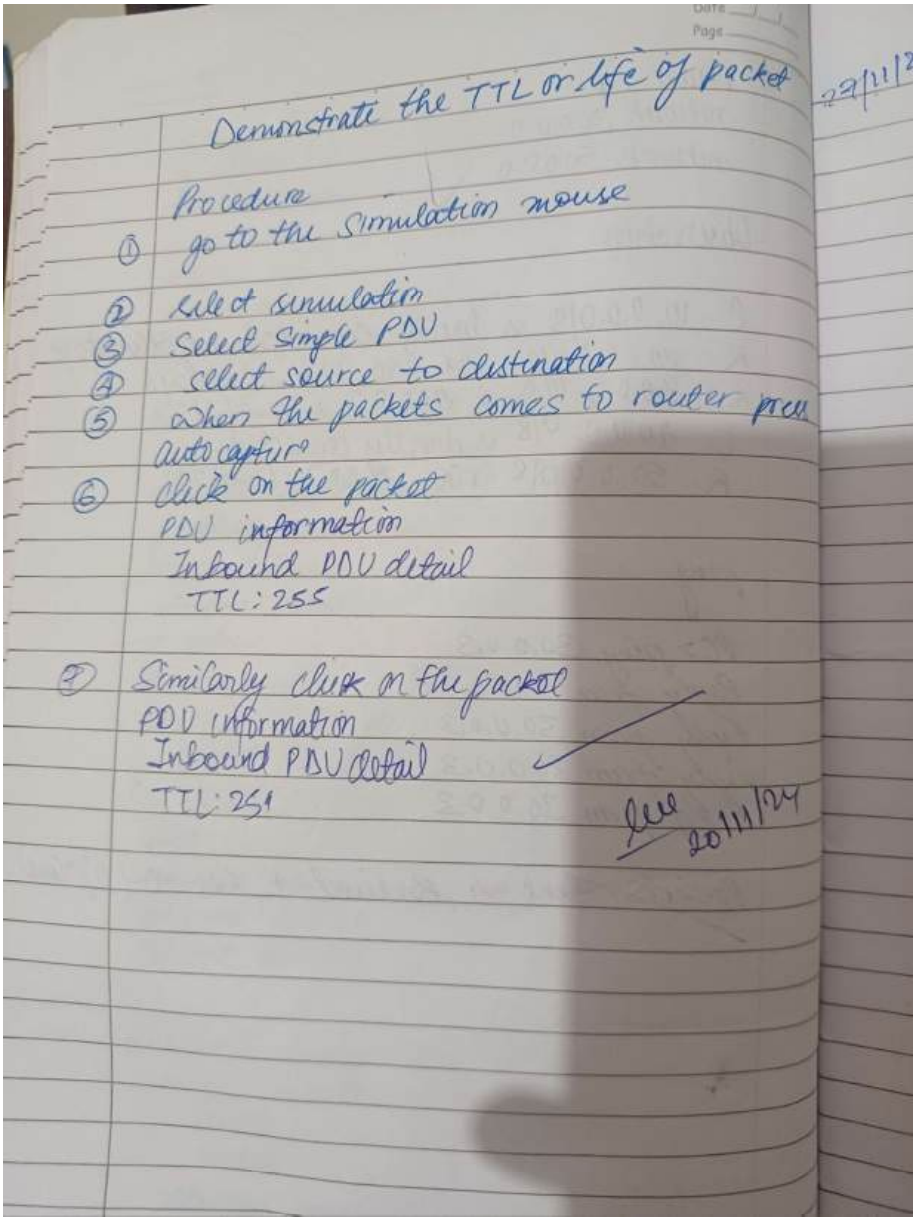
TOPOLOGY:



OUTPUT:

```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=6ms TTL=125
Reply from 40.0.0.10: bytes=32 time=4ms TTL=125

Ping statistics for 40.0.0.10:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 6ms, Average = 5ms

PC>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

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

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

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



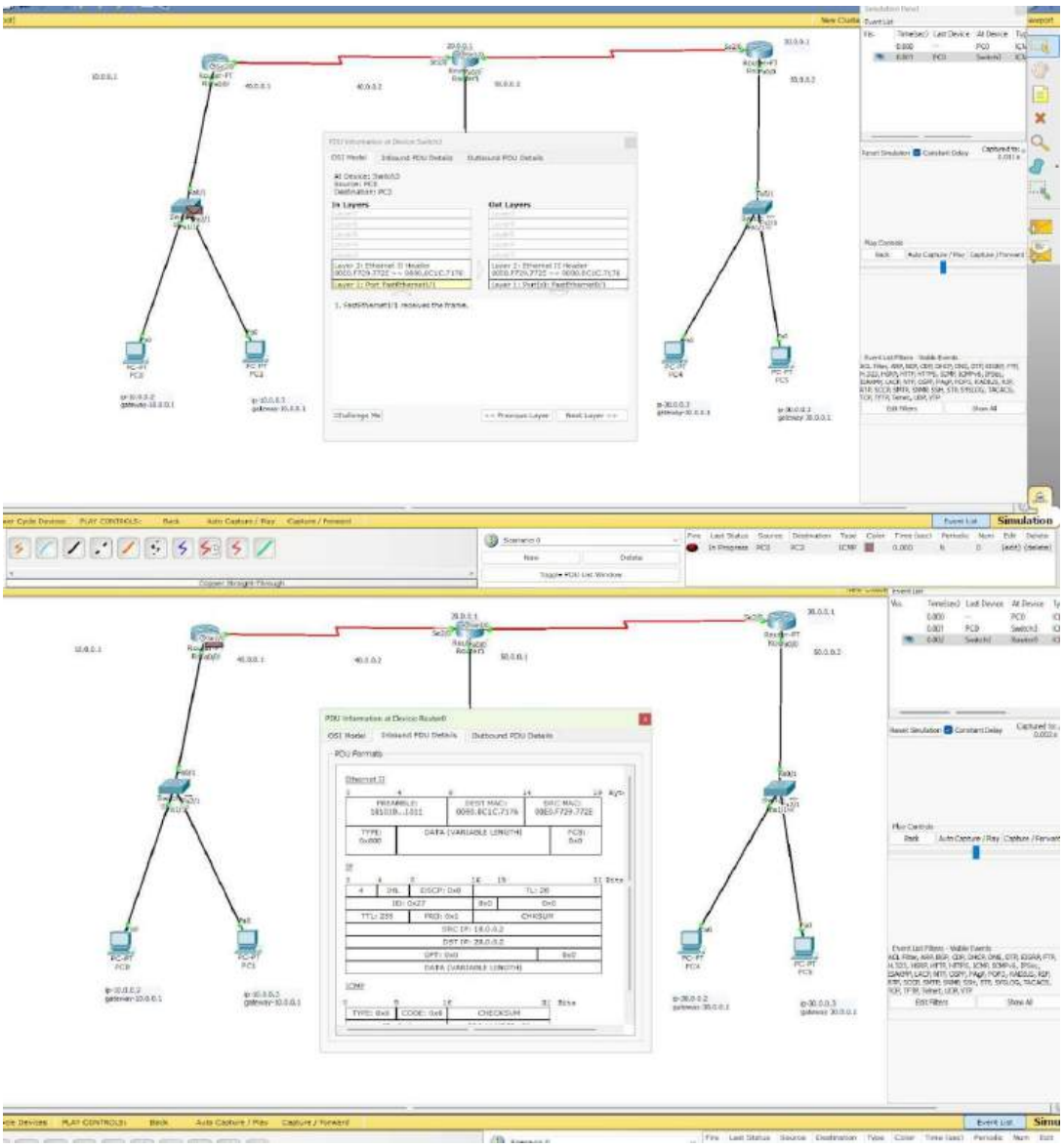
Demonstrate the TTL or life of packet 22/11/2

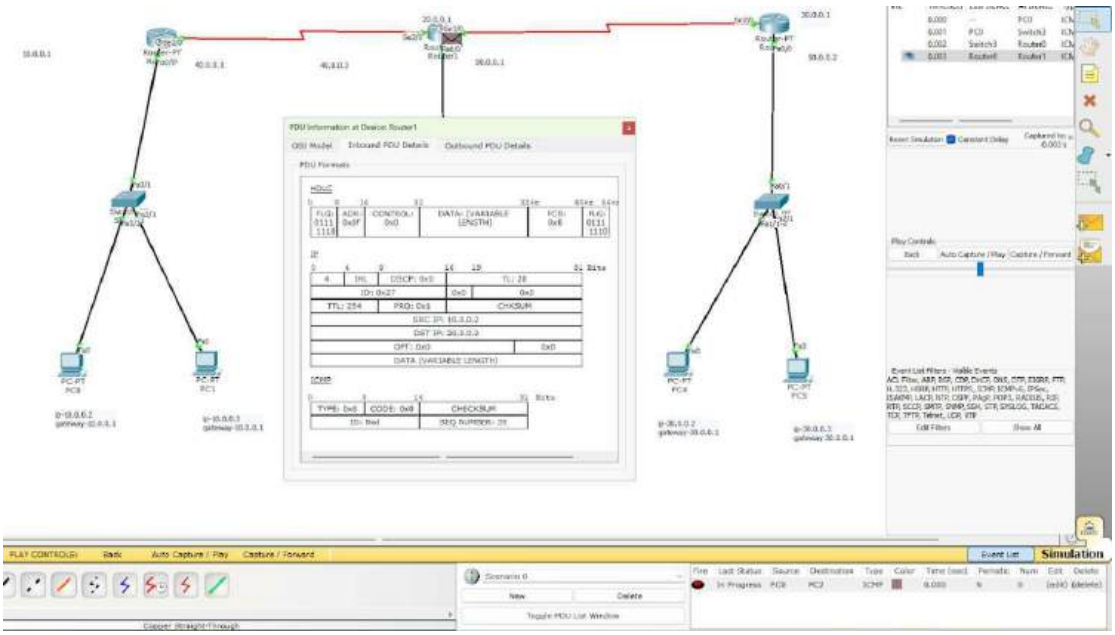
Procedure
1. go to the simulation mouse
2. select simulation
3. Select Simple PDU
4. select source to distination
5. when the packets comes to router pra auto capture
6. click on the packet
PDU information
Inbound PDU detail
TTL: 255

7. Similarly click on the packet
PDU information
Inbound PDU detail
TTL: 251

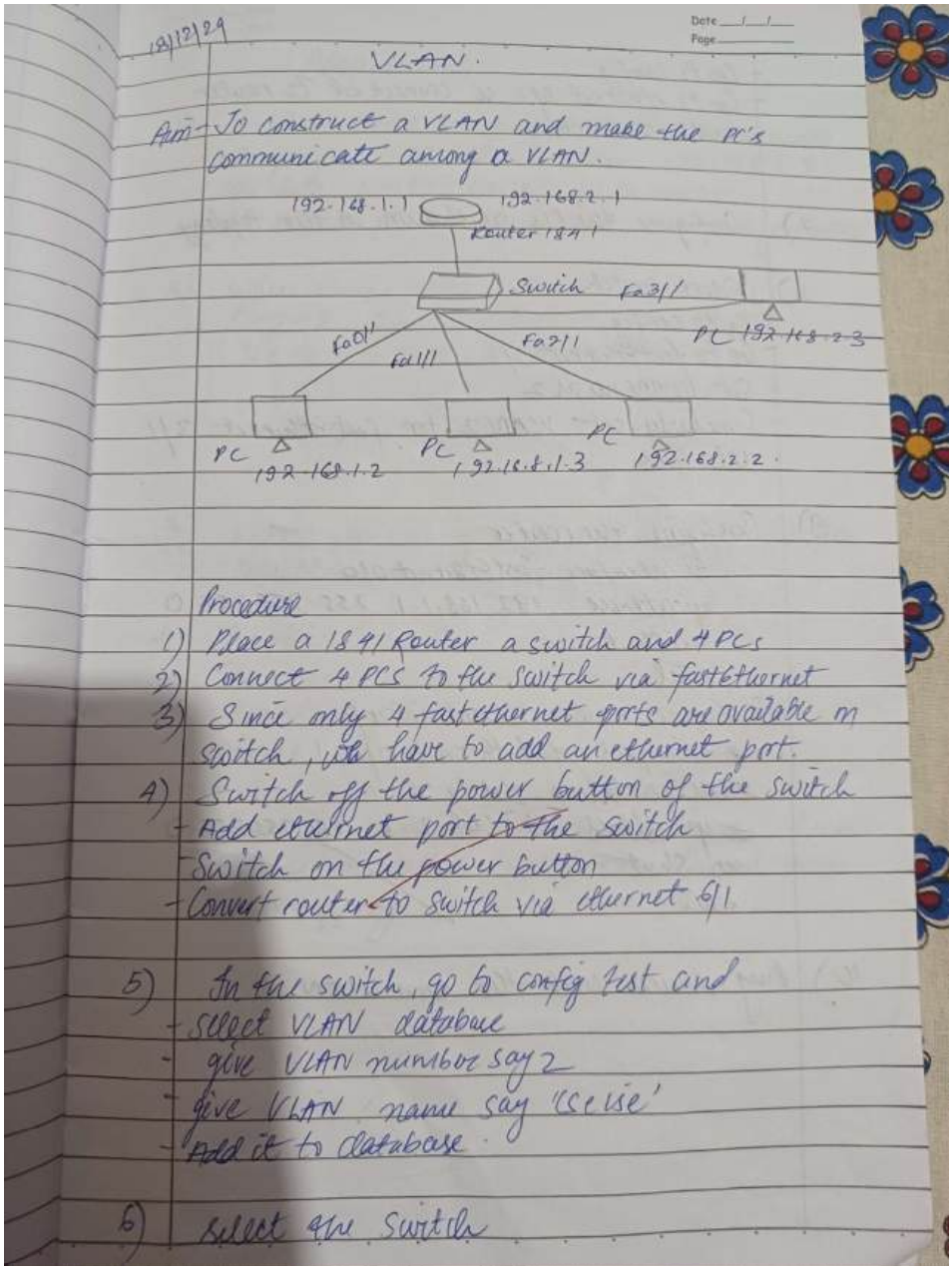
20/11/24

TOPOLOGY:





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



18/12/24

VLAN.

Aim - To construct a VLAN and make the PC's communicate among a VLAN.

192.168.1.1 192.168.2.1
Router 1841
Switch Fa3/1
Fa0/1 Fa1/1 Fa2/1
PC 192.168.2.3
PC 192.168.1.2 PC 192.168.1.3 PC 192.168.2.2.

Procedure
1) Place a 1841 Router a switch and 4 PCs
2) Connect 4 PCS to the switch via fastethernet
3) Since only 4 fastethernet ports are available on switch, we have to add an ethernet port.
4) Switch off the power button of the switch
- Add ethernet port to the switch
- Switch on the power button
- Convert router to switch via ethernet 6/1

5) In the switch, go to config tab and
- select VLAN database
- give VLAN number say 2
- give VLAN name say 'cse use'
- Add it to database.

6) Select the switch

- Go to config
- Go to ethernet 0/1 u connected to router
- Make it trunk

7) Configure the PCs as shown in the topology

8) Select switch
- Go to config
- Go to fastEthernet 2/1
- Set VLAN no as 2
- Similarly set VLAN 2 for fastEthernet 3/1 interfaces

9) Configure the router
```
# interface fastEthernet 0/0
ip address 192.168.1.1 255.255.255.0
no shut
exit
```
Configure router VLAN interface
```
# interface fastethernet 0/0.1
encapsulation dot1q 2
# ip address 192.168.2.1 255.255.255.0
no shut
exit
```

10) Ping devices with the same VLAN

Observation

1) When devices are pinged within same VLAN
- Pinging 192.168.1.3 from 192.168.1.2
- The data packet doesn't go to router.
- The switch forward without the need of router

2) When device pings a device of another VLAN
Pinging 192.168.2.3 from 192.168.1.2
The data packet journey as follows
192.168.1.2 → Switch → Router
↓
192.168.2.3 ← Switch

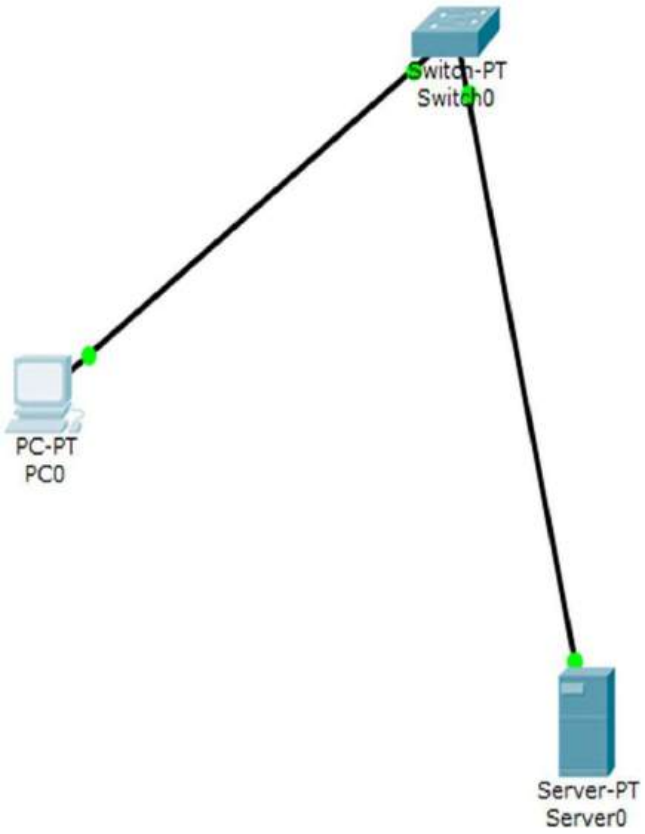
3) VLANS divide a single switch into multiple logical switches.

4) Traffic Isolation - Each VLAN maintains its own broadcast domain
- Broadcast sent by devices in one VLAN donot search devices in another VLAN

- This is done by adding additional header information called tag to the ethernet frame
VLAN tagging.

Rec

TOPOLOGY:



OUTPUT:

Quick Links:
linkdin
github
portfolio
username

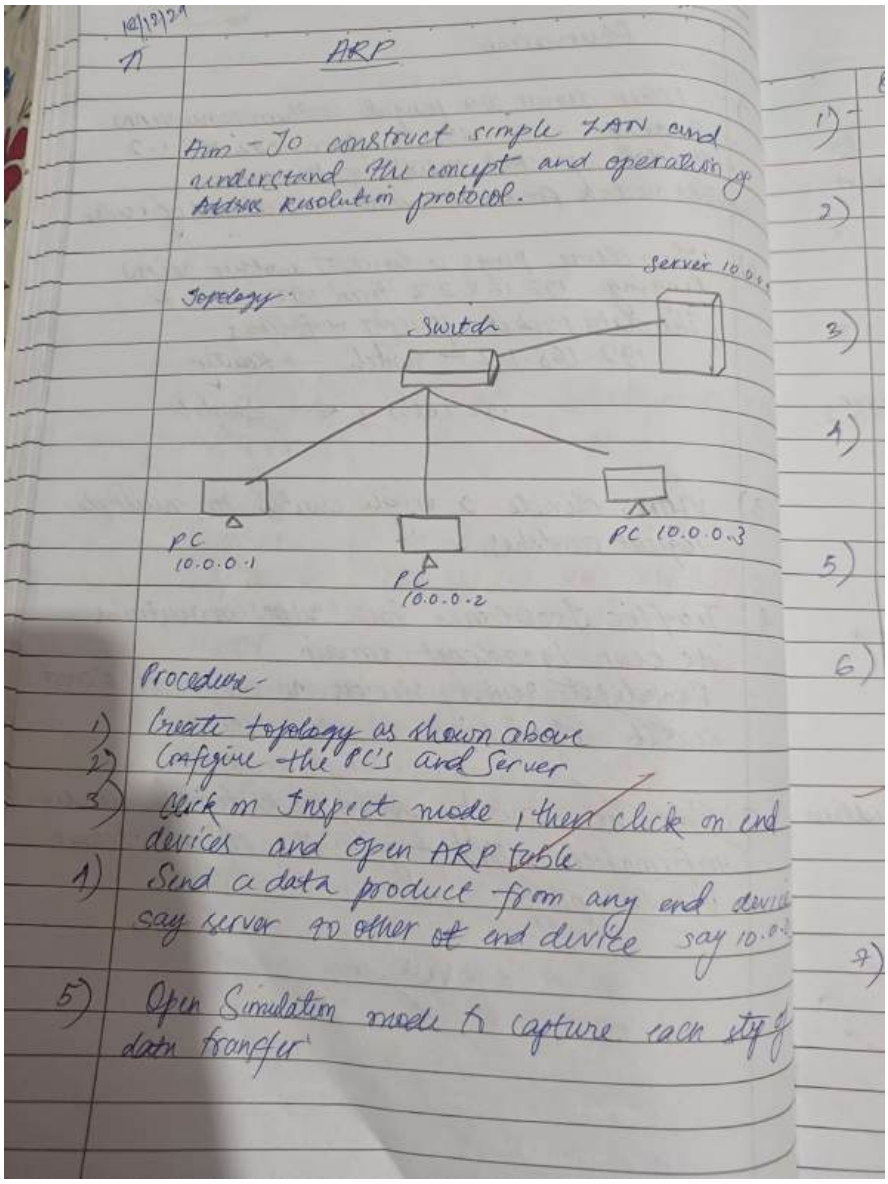
education

bms college of engineering 8.8.6 cgpa computrer science engineering

project

loibrary management system

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



ARP

Aim - To construct simple LAN and understand the concept and operation of Address Resolution protocol.

Topology:

Procedure:

1) Create topology as shown above
2) Configure the PC's and Server
3) Click on Inspect mode, then click on end devices and open ARP table
4) Send a data produce from any end device say server to other end device say 10.0.0.
5) Open Simulation mode to capture each step of data transfer

Date __/__/__
Page _____

Observation

1) The ARP tables of all end devices are initially empty

2) When the data packet from server arrives at switch. Since the MAC address is unknown, it send a broadcast signal to all devices

3) The device with IP address present in the destination address of data packet responds to message

4) The server and the PC update their ARP tables matching IP address to MAC address

5) Over time, the ARP table grows as data packets are sent.

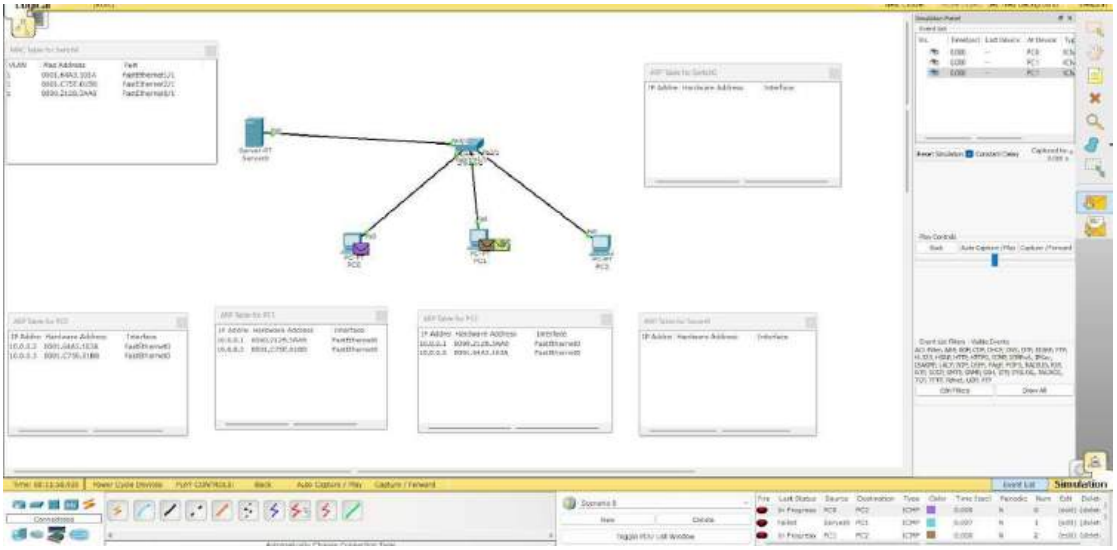
6) The MAC table of the switch which was initially empty updates it MAC gradually

ARP Table for 10.0.0.1 ☒

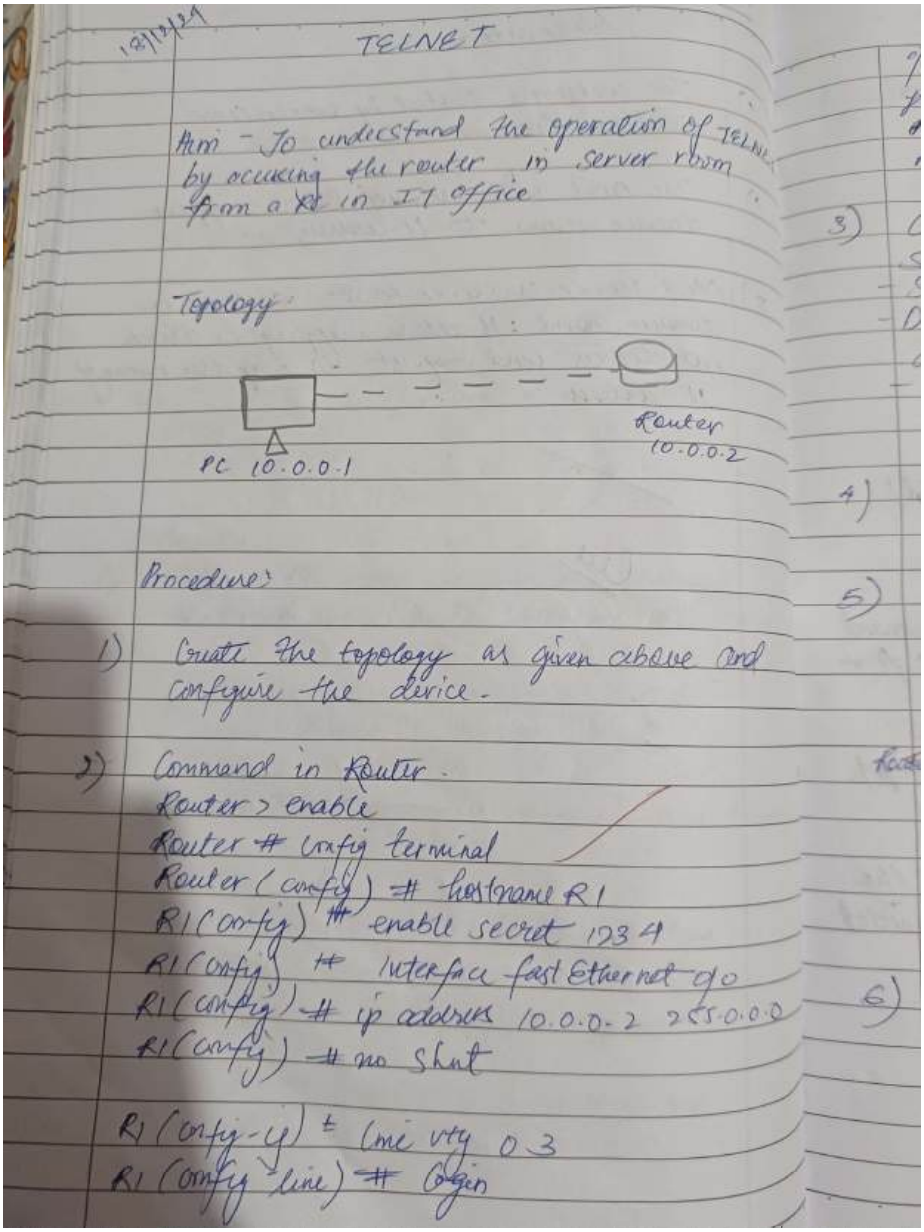
IP address	Hardware address	Interface
10.0.0.3	0001.C726.47E5	FastEthernet0

7) Similarly other ARP are updated

TOPOLOGY:



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



TELNET

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

Topology

PC 10.0.0.1 — Router 10.0.0.2

Procedure:

1) Create the topology as given above and configure the device.

2) Command in Router.
Router > enable
Router # config terminal
Router (config) # hostname R1
R1(config) # enable secret 1234
R1(config) # interface fastEthernet 0/0
R1(config) # ip address 10.0.0.2 255.0.0.0
R1(config) # no shut

R1(config-if) # line vty 0 3
R1(config-line) # login

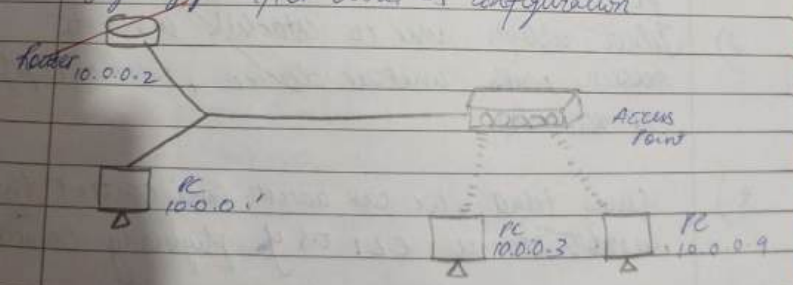
c/o login disabled on line 194, console password set.
R1(config-line)# password 4321
R1(config-line)# exit

3) Configure PC and laptop with wireless standard
- Switch off device
- Drag the existing PT-HOST-NM-1AM to the component in LHS of physical.
- Drag WMP300N wireless interface to empty port.

4) In the config tab, a new wireless interface was added

5) Configure the device by entering SSID, WEP WEP key, IP address and Gateway.

Topology after wireless configuration



6) Ping from every device to every other device to check for connection
R1(config)# exit
R1# wr
Build Configuration

3) In PC: command prompt.
- First try pinging to see if device are connected

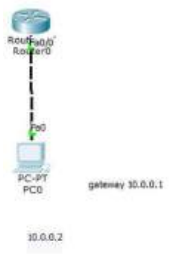
PC> telnet 10.0.0.2
Trying 10.0.0.2 open

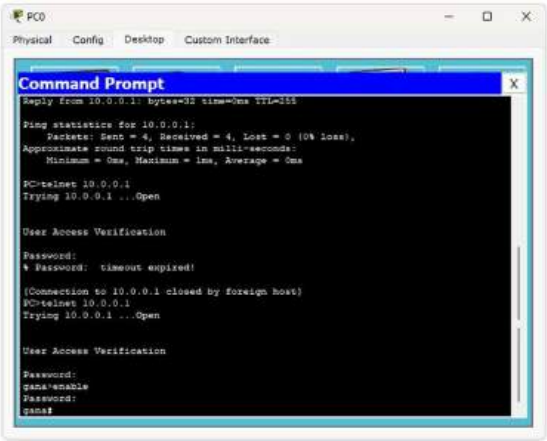
User Access verification
Password: 1321
Password: 1321
R1> enable
R1# show ip route

C 10.0.0.0/8 is directly connected, FastEthernet 0/0

Observation
1) The admin in PC is able to run command as run in router CLI and see result from PC.
2) Telnet allows user to establish a remote session with another device, over a tcp/IP network.
3) Using telnet, we can access and control the remote device CLI as if physically connected

TOPOLOGY and OUPUT:



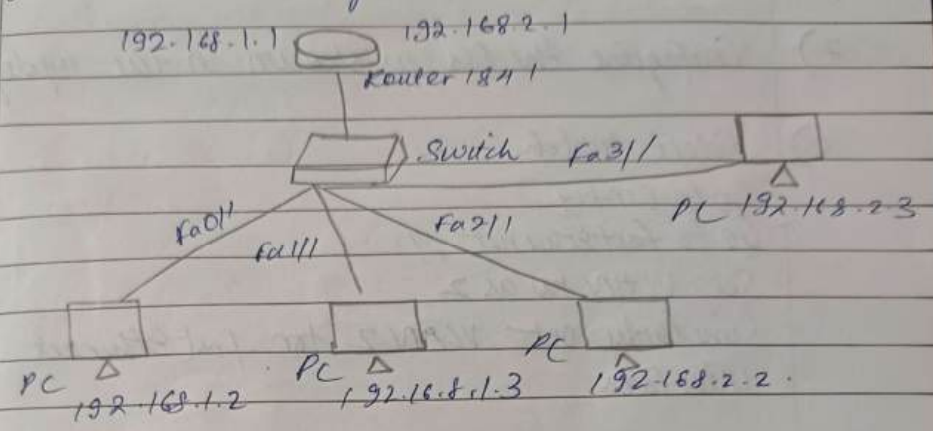


PROGRAM11: To construct a VLAN and make the PC's communicate among a VLAN

OBSERVATION:

VLAN.

Aim - To construct a VLAN and make the PC's communicate among a VLAN.



Procedure

1) Place a 1841 Router a switch and 4 PCs
2) Connect 4 PCs to the switch via fastEthernet
3) Since only 4 fastethernet ports are available on switch, we have to add an ethernet port.
4) Switch off the power button of the switch
   - Add ethernet port to the switch
   - Switch on the power button
   - Convert router to switch via ethernet 6/1

5) In the switch, go to config tab and
   - select VLAN database
   - give VLAN number say 2
   - give VLAN name say 'cse use'
   - Add it to database.

6) Select the switch

- Go to config
- Go to ethernet 6/1 ie connected to router
- Make it trunk

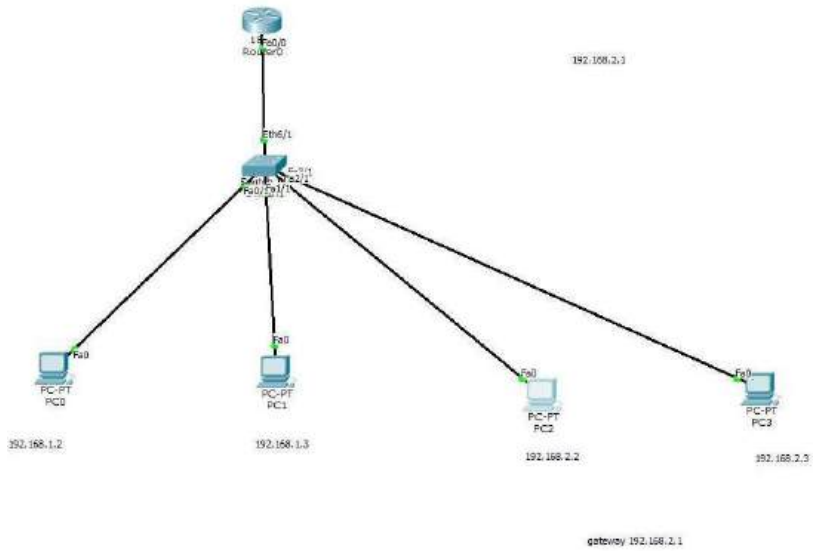
7) Configure the PCs as shown in the topology

8) Select switch
- Go to config
- Go to fastEthernet 2/1
- Set VLAN no as 2
- Similarly set VLAN 2 for fastEthernet 3/1 interface

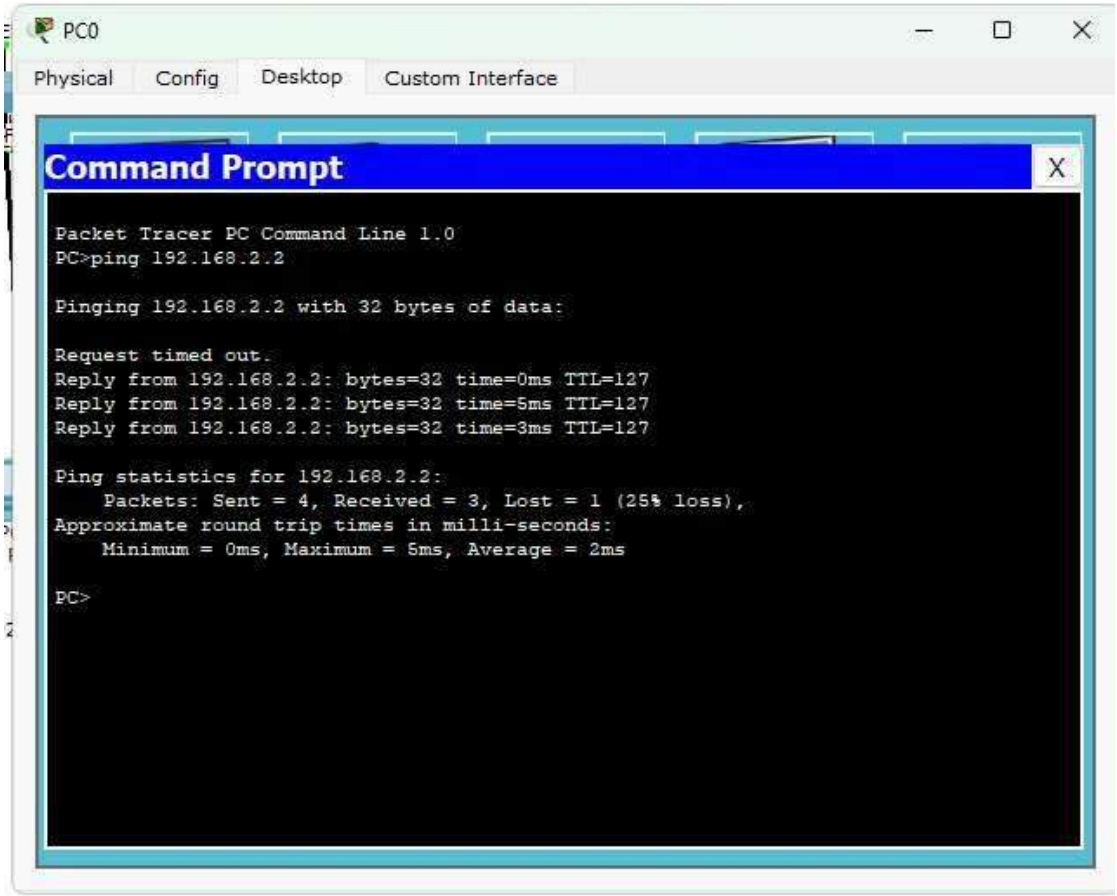
9) Configure the router
```
# interface fastEthernet 0/0
ip address 192.168.1.1 255.255.255.0
no shut
exit
```
Configure router VLAN interface
```
# interface fastethernet 0/0.1
# encapsulation dot1q 2
# ip address 192.168.2.1 255.255.255.0
no shut
exit
```

10) Ping devices with the same VLAN

TOPOLOGY:

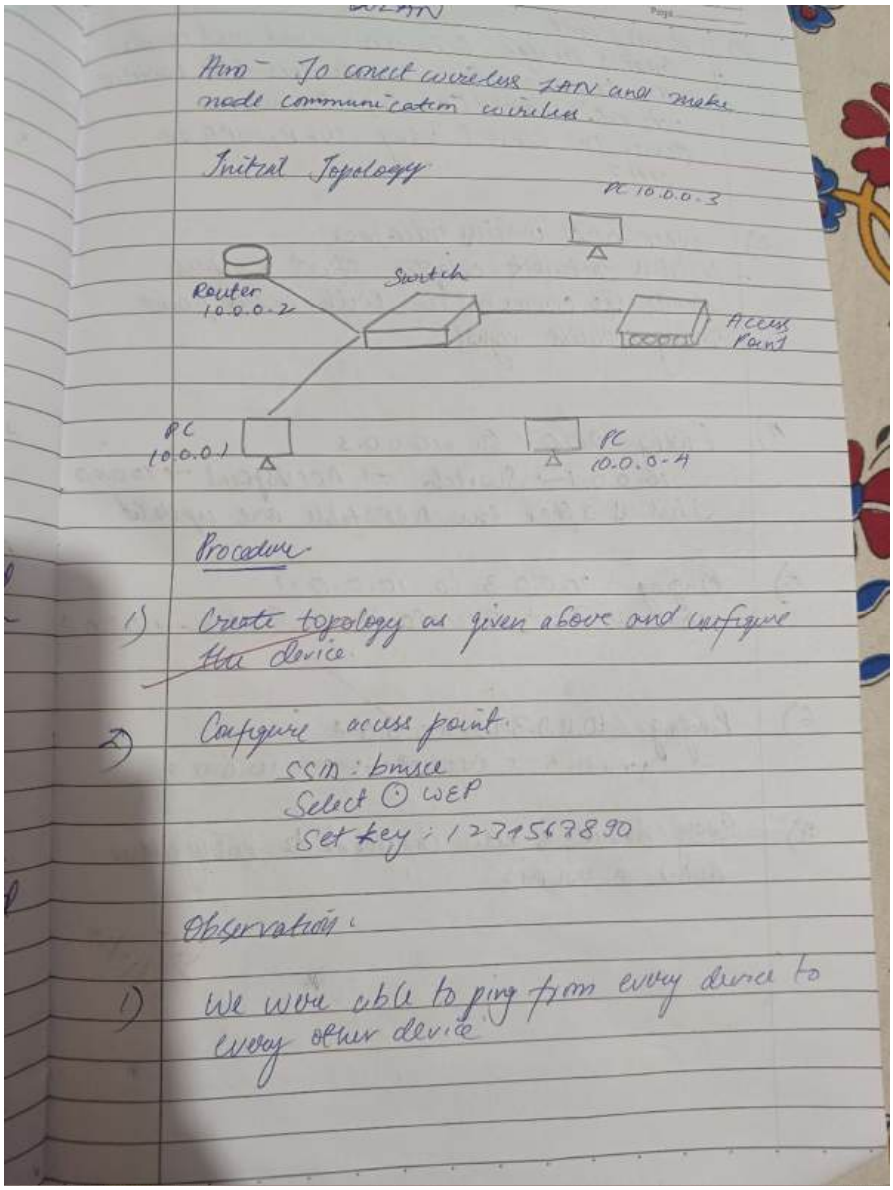


OUTPUT:



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

OBSERVATION:



Aim - To connect wireless LAN and make node communication wireless.

Initial Topology

Procedure

1) Create topology as given above and configure the device

2) Configure access point
SSID : bmsce
Select ⊙ WEP
Set key : 1234567890

Observation:

1) We were able to ping from every device to every other device

2) Access point
- Creates bridge between wired and wireless
- SSID broadcasting, announces the wireless network's name (SSID) to allow device to connect using WEP, WPA or WPA2.

3) WNR300N wireless interface
- Wireless network adapter that enables device to communicate with access point using wireless signal

4) Pinging 10.0.0.1 to 10.0.0.3
10.0.0.1 → Switch → Accesspoint → 10.
This is after the ARP table are updated

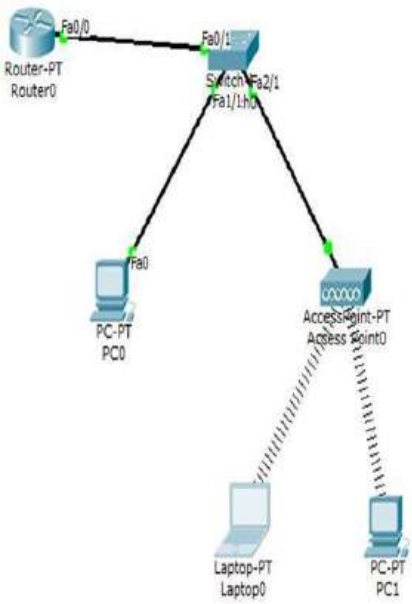
5) Pinging : 10.0.0.3 to 10.0.0.1
10.0.0.3 → AccessPoint → Switch → 10.

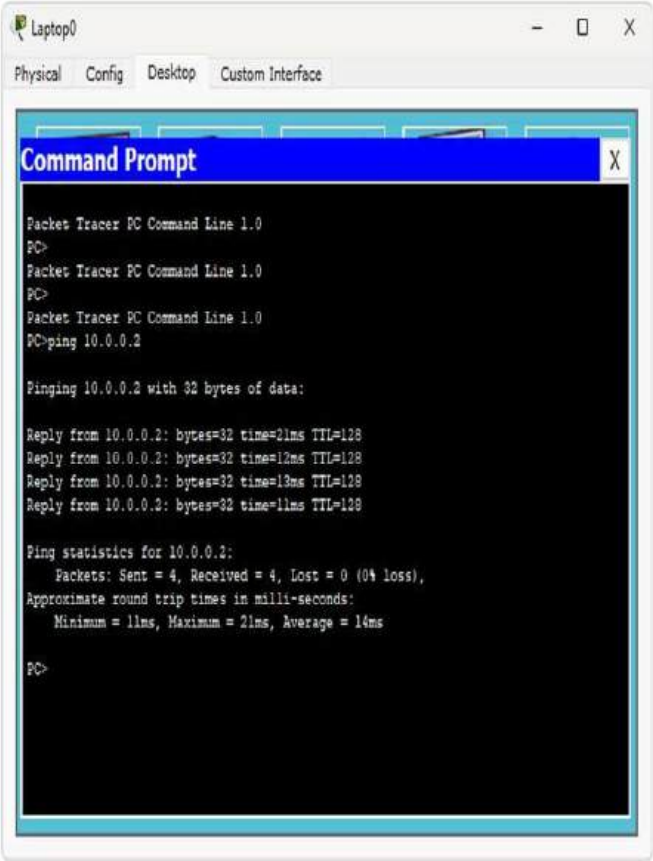
6) Pinging 10.0.0.3 to 10.0.0.1
10.0.0.3 → AccessPoint → 10.0.0.1

7) Every device is now connected to every other device in WLAN.

3/12/24

TOPOLOGY AND OUPUT:

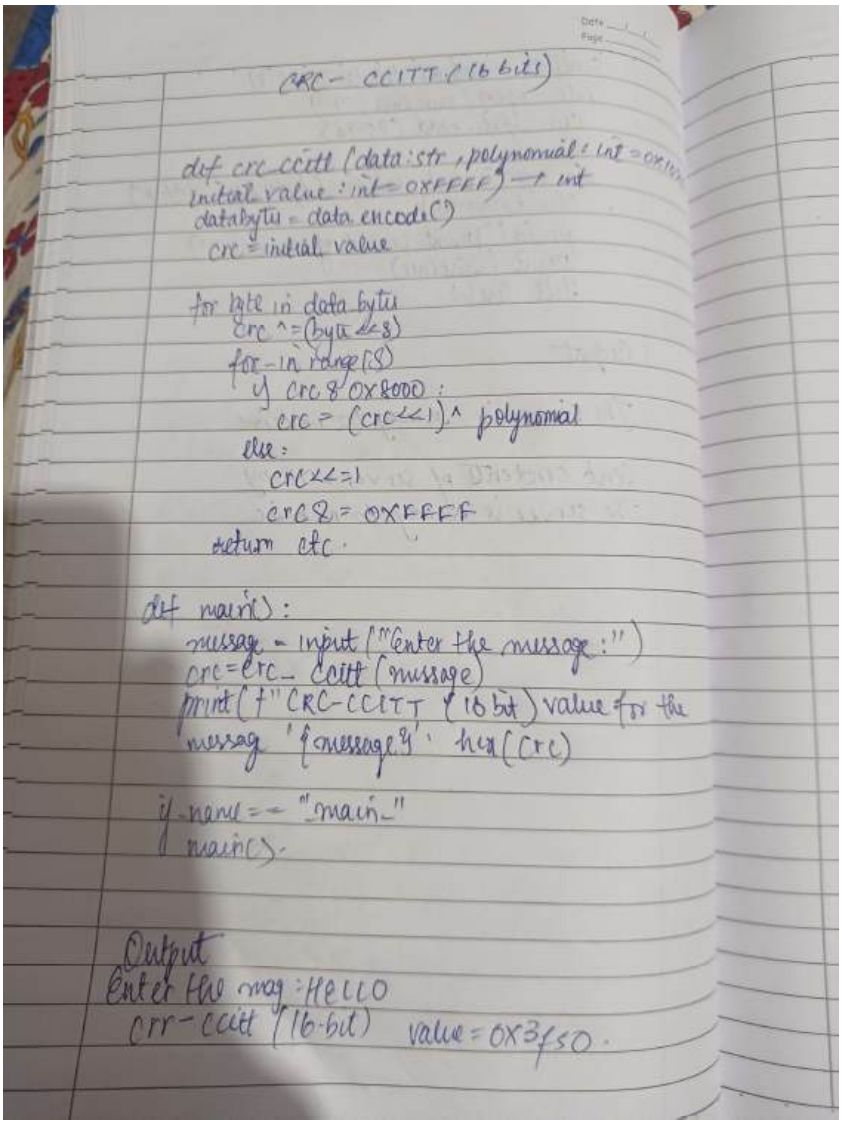




CYCLE-2

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

OBSERVATION:



```
CRC- CCITT (16 bits)

def crc_ccitt (data:str, polynomial: int = 0x1021,
initial_value: int = 0xFFFF) -> int
  databytes = data.encode()
  crc = initial_value

  for byte in data_bytes
      crc ^= (byte << 8)
      for _ in range(8)
          if crc & 0x8000 :
              crc = (crc << 1) ^ polynomial
          else:
              crc <<= 1
              crc &= 0xFFFF
      return crc.

def main():
    message = input("Enter the message:")
    crc = crc_ccitt (message)
    print(f"CRC-CCITT (16 bit) value for the
    message '{message}': hex(crc)

if __name__ == "__main__"
    main().


Output
Enter the msg: HELLO
crr-ccitt (16-bit) value = 0x3f50.
```

CODE:

```
def xor(a, b):
    result = []
    for i in range(1, len(b)):
        if a[i] == b[i]:
            result.append('0')
        else:
            result.append('1')
    return ''.join(result)


def mod2div(dividend, divisor):
    pick = len(divisor)
    tmp = dividend[0:pick]

    while pick < len(dividend):
        if tmp[0] == '1':
            tmp = xor(divisor, tmp) + dividend[pick]
        else:
            tmp = xor('0' * pick, tmp) + dividend[pick]
        pick += 1

    if tmp[0] == '1':
        tmp = xor(divisor, tmp)
    else:
        tmp = xor('0' * pick, tmp)

    checkword = tmp
    return checkword


def encode(data, key):
    key_len = len(key)
    appended_data = data + '0' * (key_len - 1)
    remainder = mod2div(appended_data, key)
    codeword = data + remainder
    print(f"Encoded  Data:  {codeword}")
    return codeword


def decode(data, key):
    remainder = mod2div(data, key)
    print(f"Remainder after decoding: {remainder}")
    if '1' not in remainder:
        print("No error detected in received data")
    else:
        print("Error detected in received data")


# Main function
```

```
if __name___== "__main__":
    data = input("Enter the  data bits:  ")
    key = input("Enter the key (divisor): ")

    # Encoding
    encoded_data = encode(data, key)

    # Decoding
    print("\nDecoding the encoded data...")
    decode(encoded_data, key)
```

OUTPUT:

```
Enter the data bits: 111100000111010
Enter the key (divisor): 1010111
Encoded Data: 111100000111010110101

Decoding the encoded data...
Remainder after decoding: 000000
No error detected in received data

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

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

OBSERVATION:

```
int main()
    int packet_sz[NOF_PACKETS], i, clk, b_size
    o_rate, p_sz_rm=0, p_sz, p_time, op;
    for(int i=0; i<NOF_PACKETS; i++)
        packet_sz[i] = random%100;
    for(int i=0; i<NOF_PACKETS; i++){
        printf("\n packet [%d] %d bytes", i,
          packet_sz[i]);
        printf("Enter output rate");
        scanf("%d", &b_size);
        printf("Enter the bucket size");
        scanf("%d", &b_size);
    for(i=0; i<NOF_PACKETS; i++){
        if(packet_sz[i]+p_sz_rm)>b_size)
          if(packet_sz[i]>b_size)
            printf("\n\n Incoming packet size
              is greater than bucket size")

        else {
          p_sz_rm += packet_sz[i];
          printf("\n Incoming packet, packet_sz[i]);
          printf("Byte remaining to Transmit, p_sz_rm);
          while(p_sz_rm)
            sleep(1);
              if(p_sz_rm <= o_rate)
              op=p_sz_rm; p_sz_rm=0;
              else
                op=o_rate, p_sz_rm= o_rate;
                printf("Packet size", op);
                printf("Bytes remaining to transmit");
          }
          else{ printf("No packets to transmit");
        }
```

CODE:

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

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

    // Input number of packets and their sizes
    for(i = 0; i < NOF_PACKETS; i++) {
        packet_size[i] = rand() % 100;  // Random packet size between 0 and 99
```

```
        printf("packet[%d]:%d bytes\n", i, packet_size[i]);
    }

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

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

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

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

OUTPUT:

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

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

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

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

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

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


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

PROGRAM3: Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.
OBSERVATION:

Client Server program using
TCP, IP sockets

Client-TCP.py

```
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket (AF_INET, SOCK_STREAM)
clientSocket.connect ((serverName, serverPort))
sentence = input ("Enter file name")

clientSocket.send (sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print("from Server");
print (filecontents)
clientSocket.close()
```

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

connectionSocket.send(l.encode())
print("In Sent contents of" + sentence)
file.close()
connectionSocket.close()
```

Output

```
The server is ready to receive
Sent contents of serverTCP.py
The server is ready to receive
```

CODE:

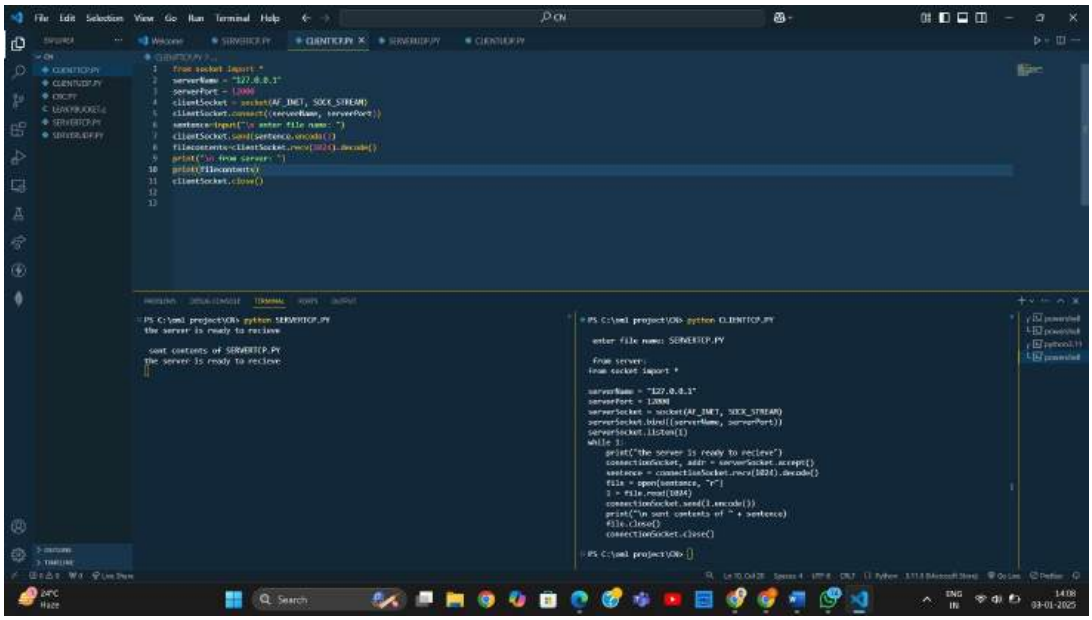
SERVERTCP.PY:

```
from socket import *

serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while 1:
    print("the server is ready to recieve")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file = open(sentence, "r")
    l = file.read(1024)
    connectionSocket.send(l.encode())
    print("\n sent contents of " + sentence)
    file.close()
    connectionSocket.close()
```

CLIENTTCP.PY:

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

OUTPUT:


File Edit Selection View Go Run Terminal Help
CLIENTTCP.PY
SERVERTCP.PY
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
sentence=input("\n enter file name: ")
clientSocket.send(sentence.encode())
filecontents=clientSocket.recv(1024).decode()
print("\n from server: ")
print(filecontents)
clientSocket.close()
PS C:\oml project\OR> python SERVERTCP.PY
the server is ready to recieve
sent contents of SERVERTCP.PY
the server is ready to recieve
PS C:\oml project\OR> python CLIENTTCP.PY
enter file name: SERVERTCP.PY
from server:
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while 1:
print("the server is ready to recieve")
connectionSocket, addr = serverSocket.accept()
sentence = connectionSocket.recv(1024).decode()
file = open(sentence, "r")
l = file.read(1024)
connectionSocket.send(l.encode())
print("\n sent contents of " + sentence)
file.close()
connectionSocket.close()
PS C:\oml project\OR>

PROGRAM4: Using UDP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.
OBSERVATION:

Client server using
UDP socket

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, serverAdd = clientSocket.recvfrom(
print ("\n Reply from Server
print (fileContents.decode("Utf-8"))

clientSocket.close()
clientSocket.close()
```

ServerUDP.py

```
from socket import *
ServerPort = 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 ("Insent content of ", end = ' ')
print (sentence)
file.close()
```

Output

The server is ready to receive

Sent contents of serverUDP.py
The server is ready to receive

CODE:
SERVERUDP.PY

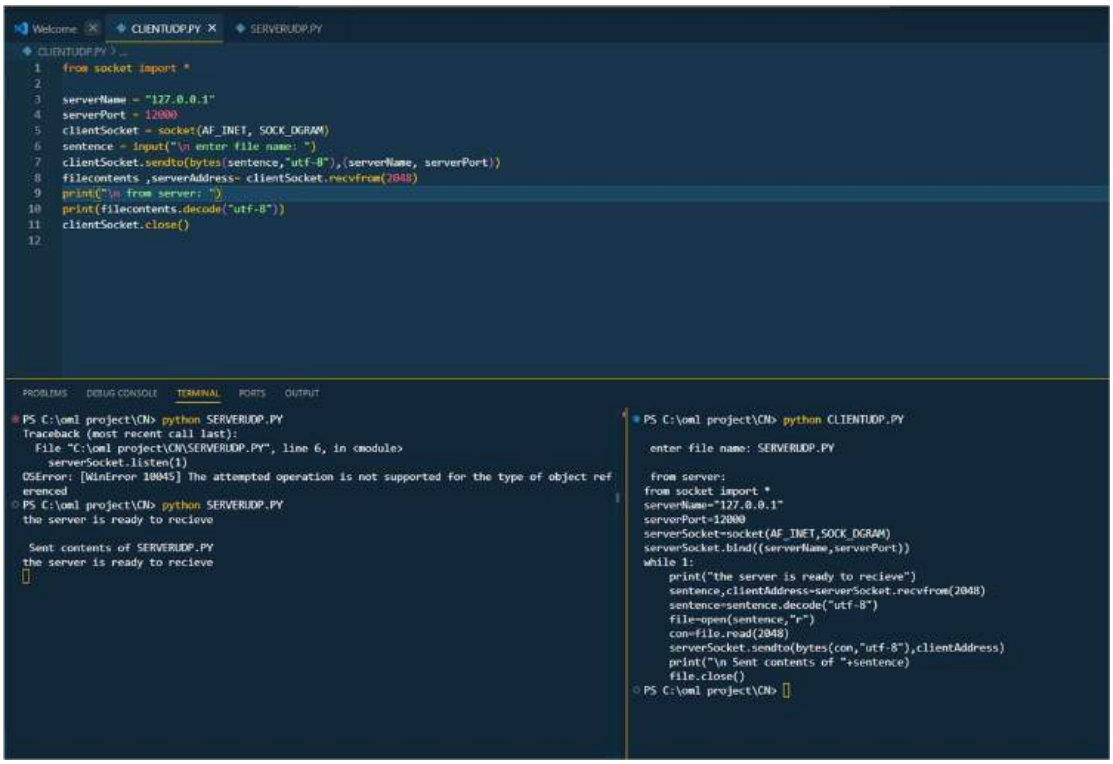
```
from socket import *
serverName="127.0.0.1"
```

```
serverPort=12000
serverSocket=socket(AF_INET,SOCK_DGRAM)
serverSocket.bind((serverName,serverPort))
while 1:
    print("the server is ready to recieve")
    sentence,clientAddress=serverSocket.recvfrom(2048)
    sentence=sentence.decode("utf-8")
    file=open(sentence,"r")
    con=file.read(2048)
    serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
    print("\n Sent contents of "+sentence)
    file.close()
```

CLIENTUDP.PY:

```
from socket import *

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

OUTPUT:



WIRESHARK:

Tool Exploration - Wireshark

Wireshark is a powerful and widely used network protocol analyzer. It allows you to inspect and capture data packets travelling over a network in real time making it a crucial tool for studying computer network

Key feature

1. Packet capture, capture live network traffic from various interface

2. Protocol Analysis - Supports hundreds of protocol TCP, UDP, HTTP, FTP

3. Filterings offers powerful filters to isolate specific packet or trafic

4. Visualization - Display packet details with heirachal layers.

Use case of wireshark
Network troubleshooting
- Diagnosing slow network speed
- Identifying bottle neck misconfiguration

2. Security Analysis
malicious traffic or intrusion

Date __/__/__
Page _____

8 Protocol Study
Understanding packet structures and communication flow

Common filters

http show only HTTP traffic
tcp.port=80: show traffic on TCP port 80
IP add = 192.168.1.1 : show packet from specific IP address
UDP: show only UDP traffic.