VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB RECORD

Computer Networks

Submitted by

Ayman Khan (1BM22CS062)

in partial fulfilment 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
Academic Year 2024-25 (odd)

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 Network (23CS5PCCON)" carried out by **Ayman Khan (1BM22CS062),** who is a bonafide student of **B.M.S. College of Engineering.** It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum. The Lab report has been approved as it satisfies the academic requirements of the above-mentioned subject and the work prescribed for the said degree.

Dr. Nandhini Vineeth Associate Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor & HOD Department of CSE, BMSCE

Index

Cycle - I

Cycle-II

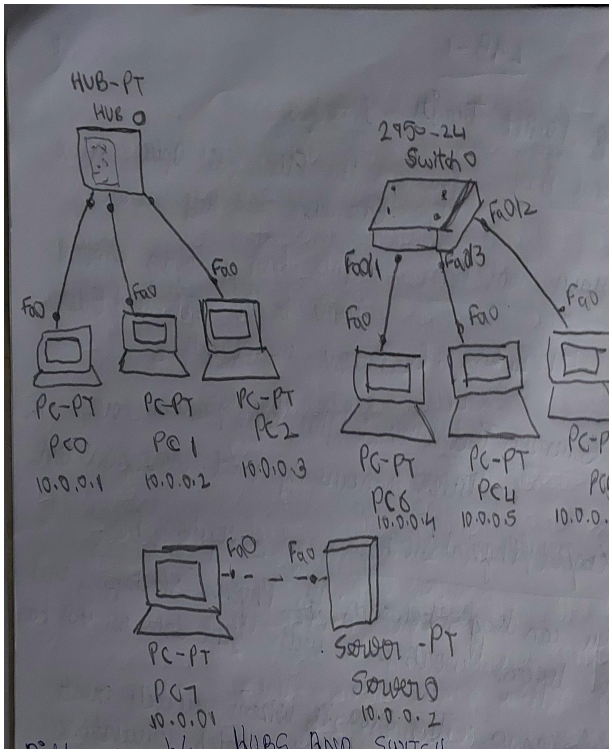
Github Link:
https://github.com/AymanKhan97/CN-LAB

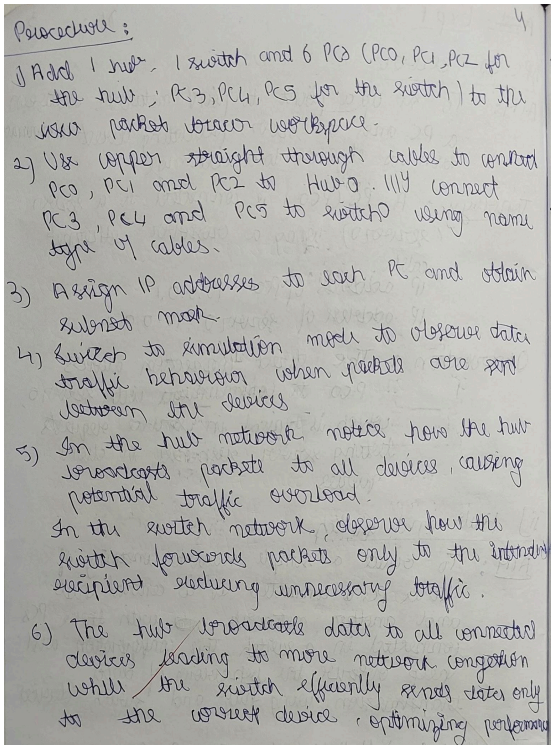
Program 1

Aim of the program:

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

Procedure along with the topology:

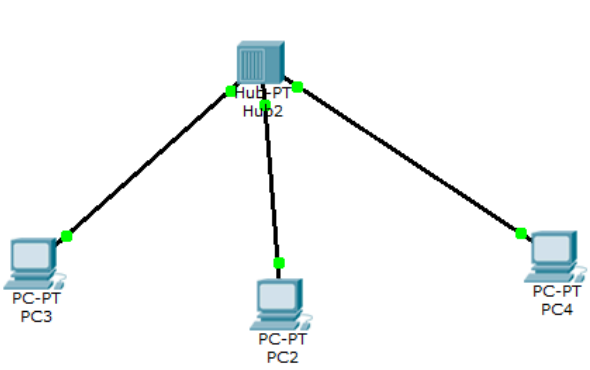


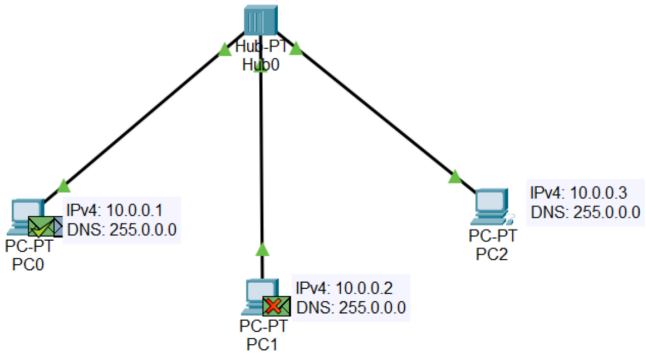


Procedure:

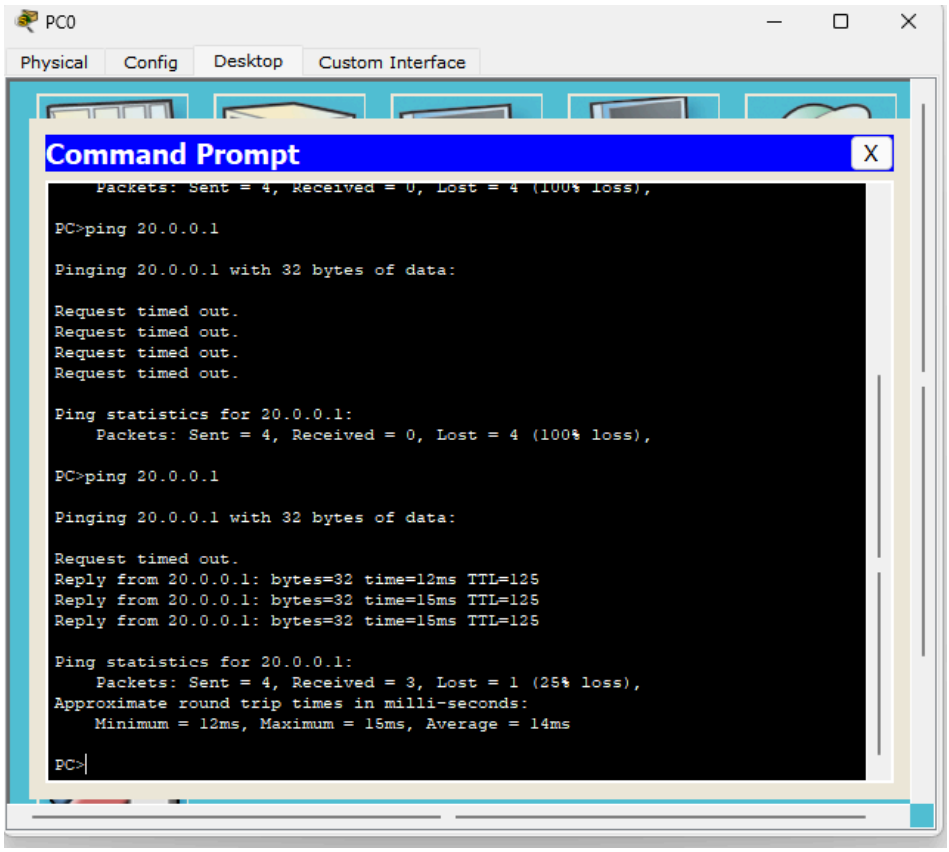
1) Add 1 hub, 1 switch and 6 PCs (PC0, PC1, PC2 for the hub ; PC3, PC4, PC5 for the switch) to the cisco packet tracer workspace.
2) Use copper straight through cables to connect PC0, PC1 and PC2 to Hub 0. Next connect PC3, PC4 and PC5 to switch0 using same type of cables.
3) Assign IP addresses to each PC and obtain subnet mask.
4) Switch to simulation mode to observe data traffic behaviour when packets are sent between the devices.
5) In the hub network notice how the hub broadcasts packets to all devices, causing potential traffic overload.
In the switch network, observe how the switch forwards packets only to the intended recipient reducing unnecessary traffic.
6) The hub broadcasts data to all connected devices leading to more network congestion while the switch efficiently sends data only to the correct device, optimizing performance

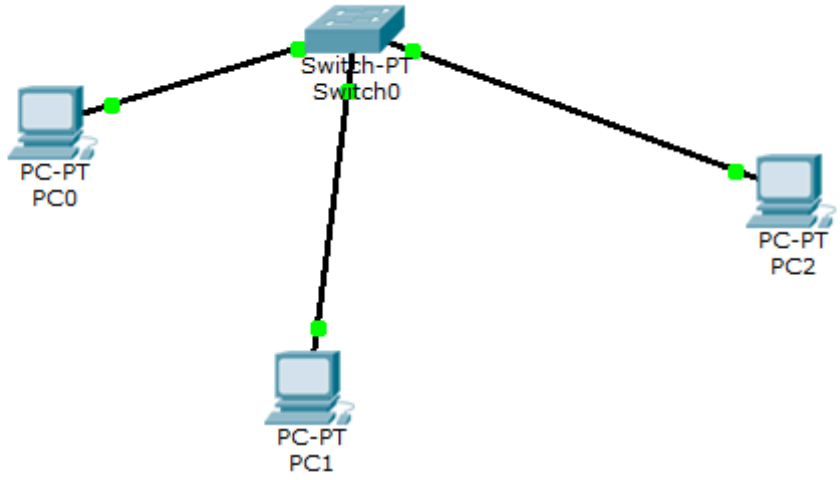
Screen shots/ output:





Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC2	ICMP		0.000	N	0	(edit)	

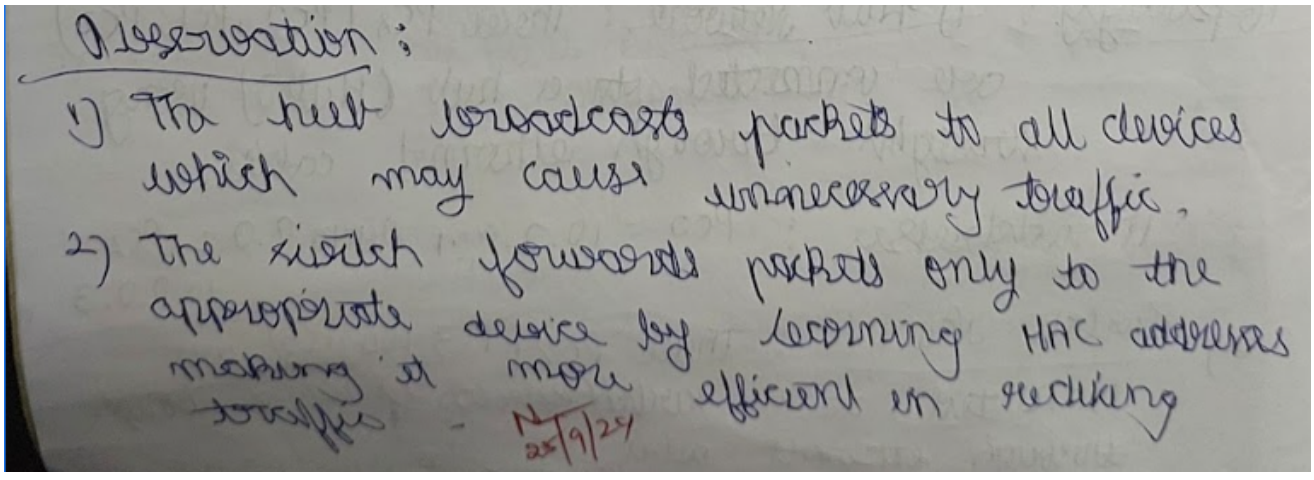




Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC1	ICMP		0.000	N	0	(edit)	(delete)

```
C:\>ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data:

Reply from 10.0.0.3: bytes=32 time=9ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms TTL=128
Reply from 10.0.0.3: bytes=32 time=1ms TTL=128
Reply from 10.0.0.3: bytes=32 time<1ms 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 = 9ms, Average = 2ms
```

Observation:

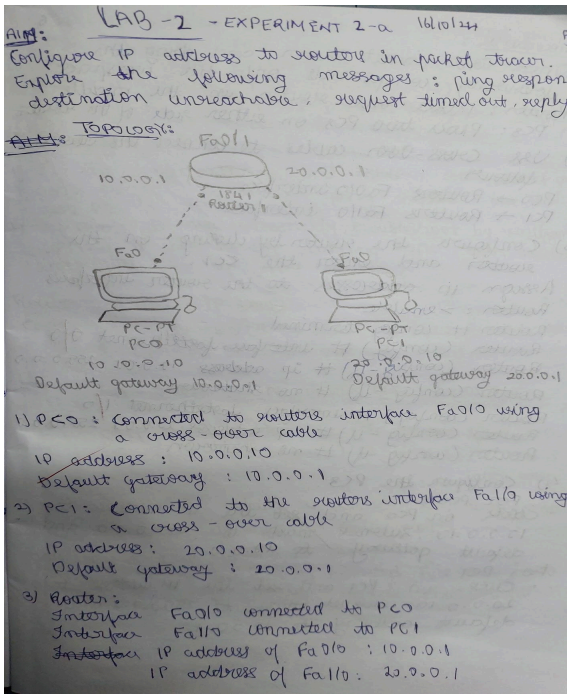


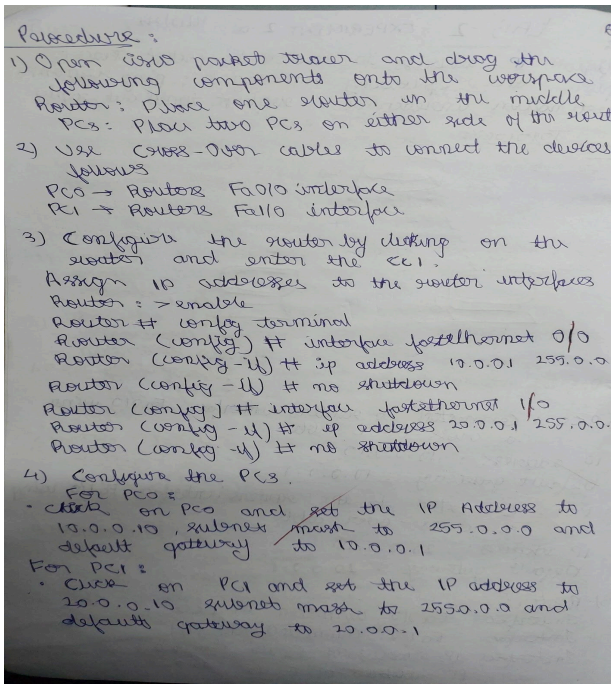
Program 2-a

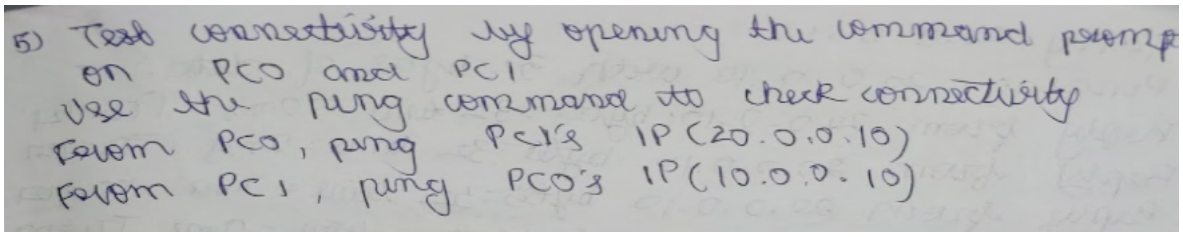
Aim of the program:

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

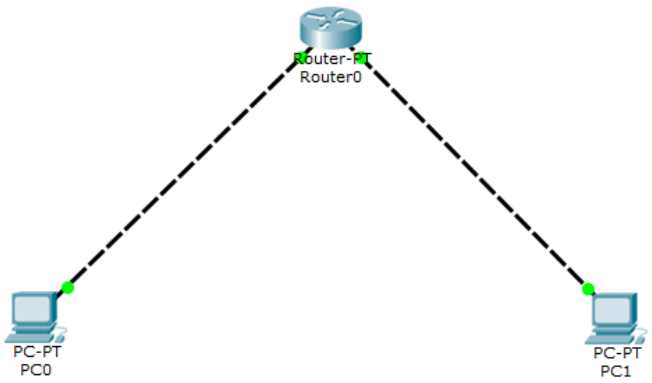
Procedure along with the topology:

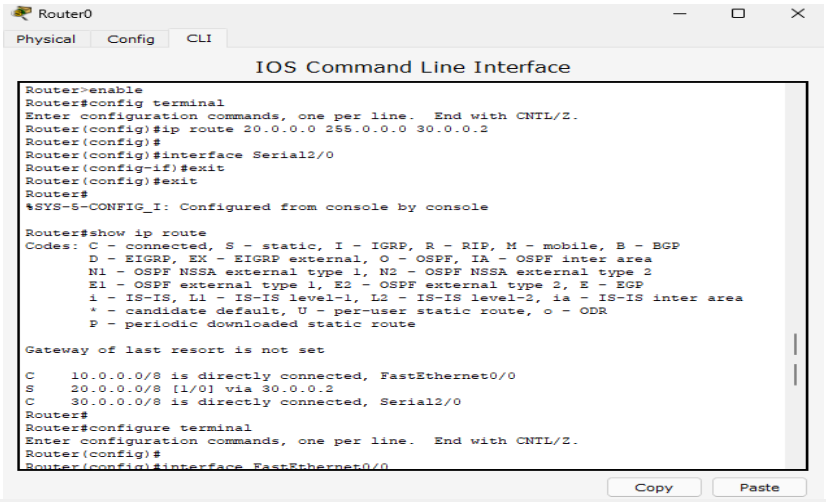


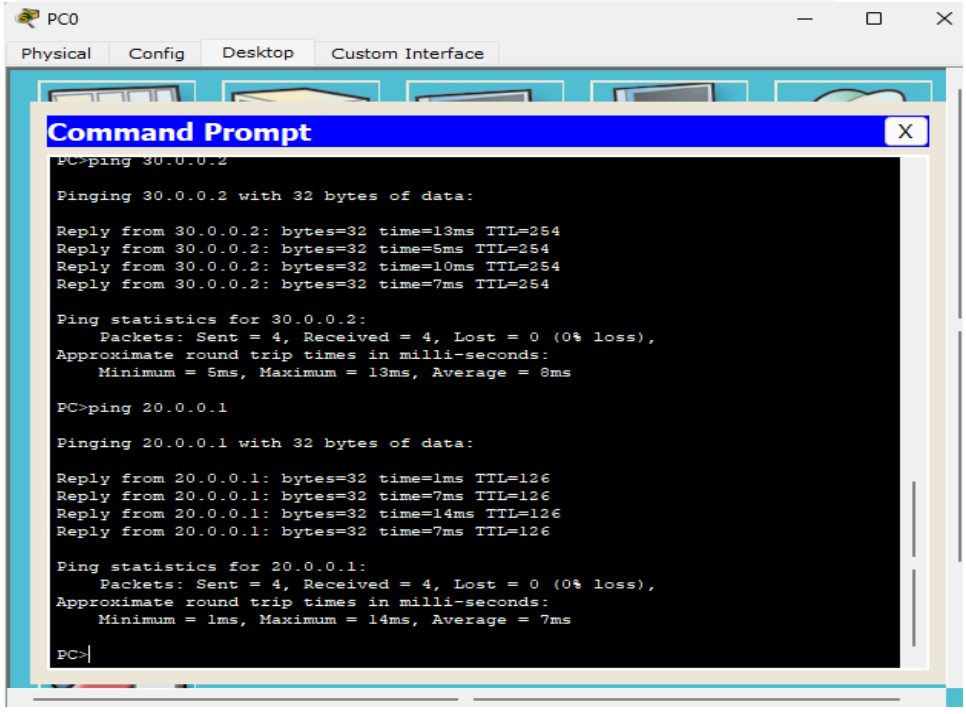




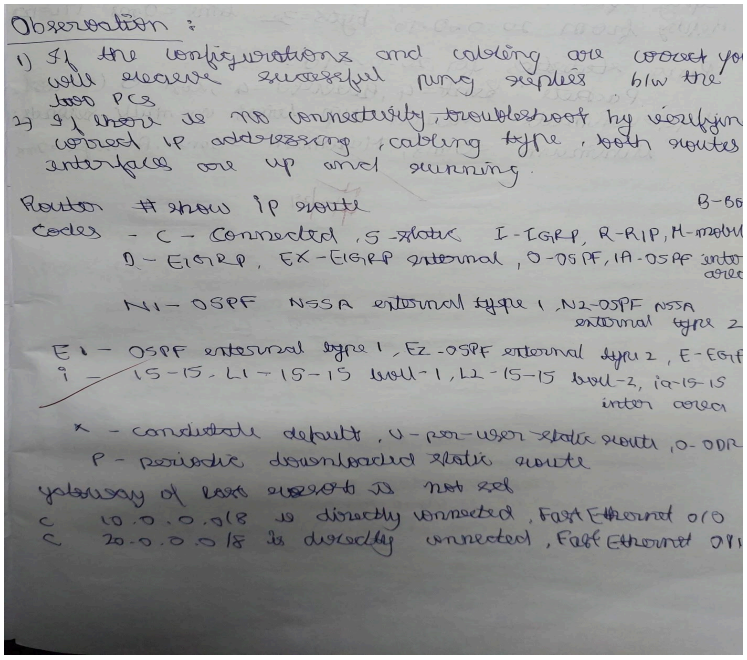
Screen shots/ output:







Observation:



Observation :

1) If the configurations and cabling are correct you will receive successful ping replies b/w the two PCs
2) If there is no connectivity, troubleshoot by verifying correct IP addressing, cabling type, both routes interfaces are up and running.

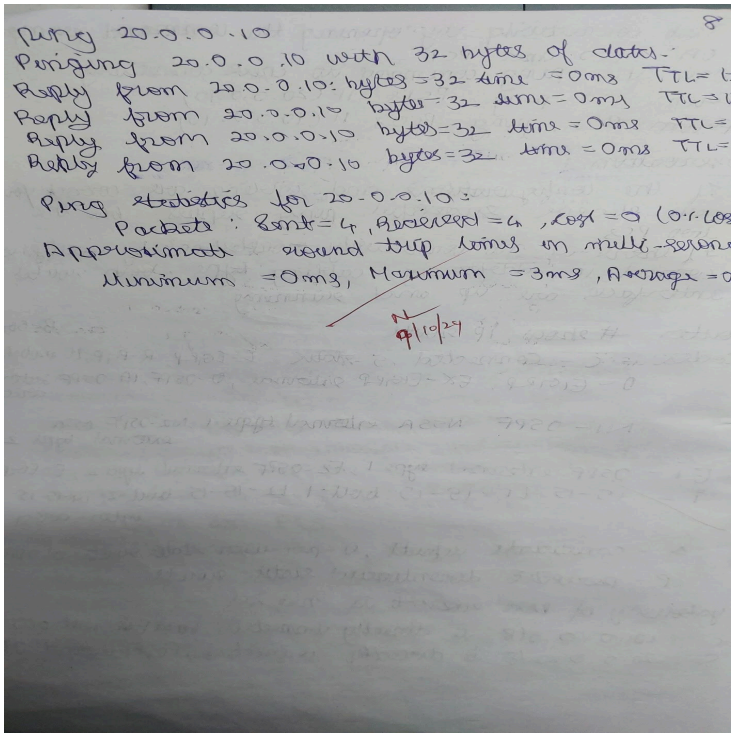
Router # show ip route

Codes - C - Connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP, D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area, N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2, E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP, i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area, * - candidate default, U - per-user static route, o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

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

C 20.0.0.0/8 is directly connected, FastEthernet 0/1



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=

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

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

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

Ping statistics for 20.0.0.10:

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

Approximate round trip times in milli-seconds:

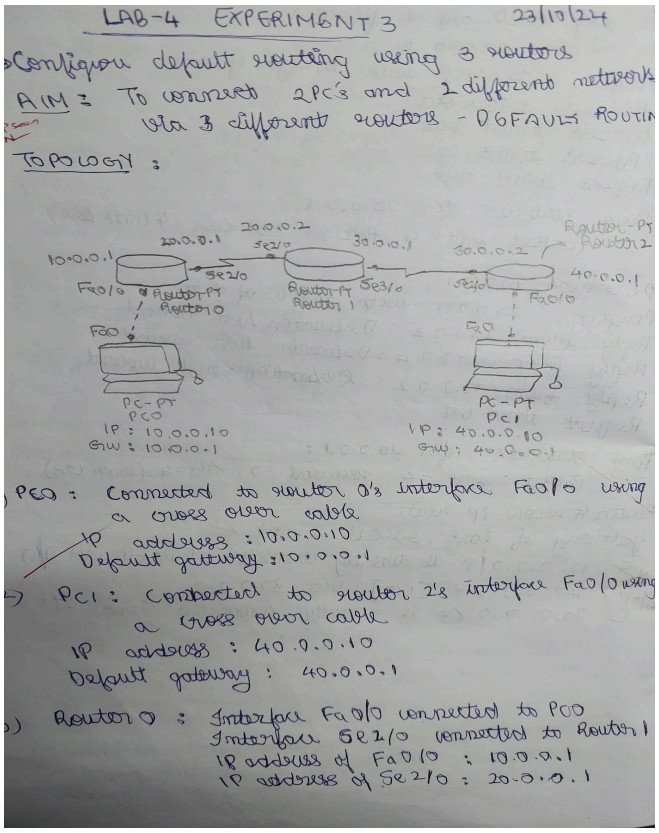
Minimum =0ms, Maximum =3ms, Average =0

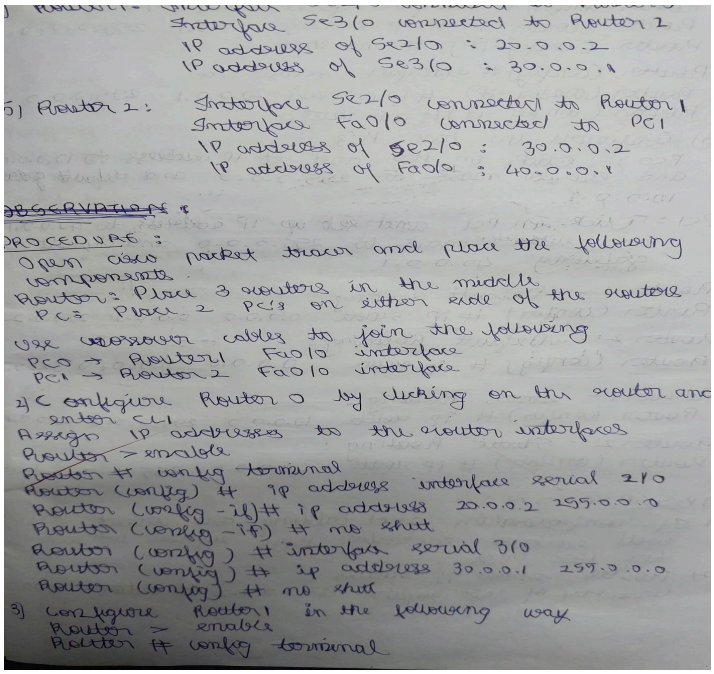
9/10/24

Program 3

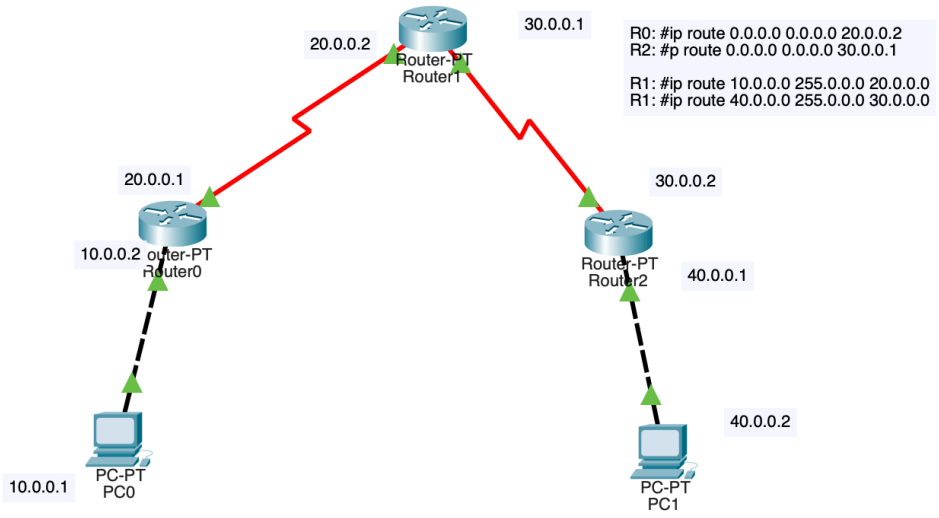
Aim of the program:
Configure default route, static route to the Router

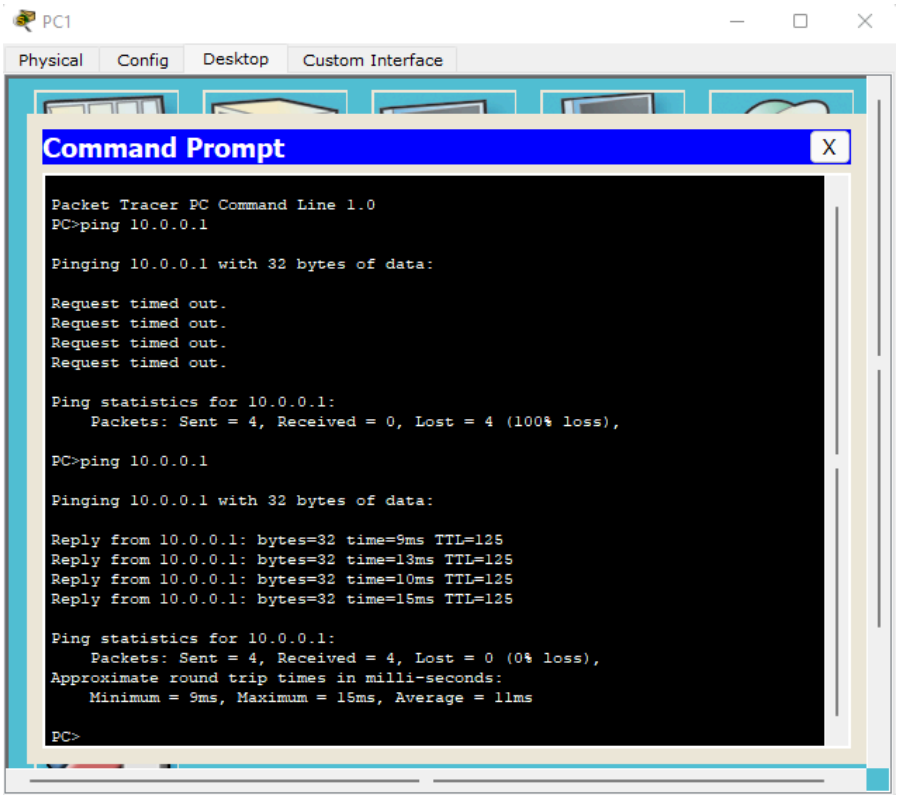
Procedure along with the topology:

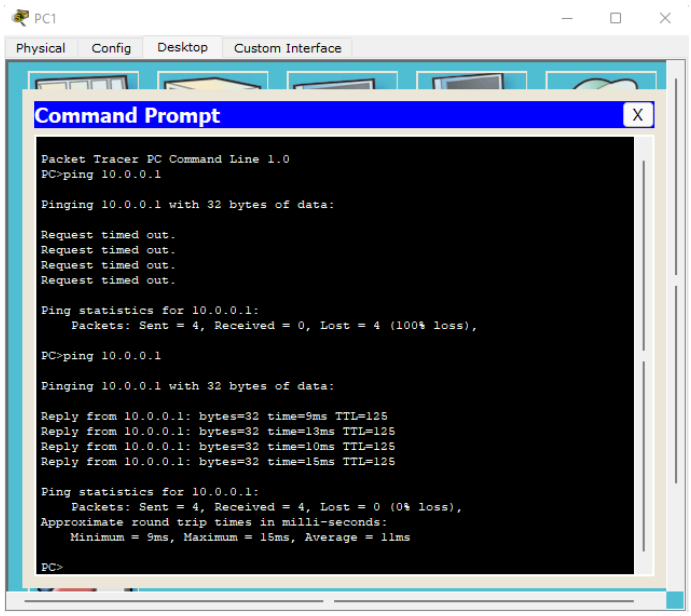


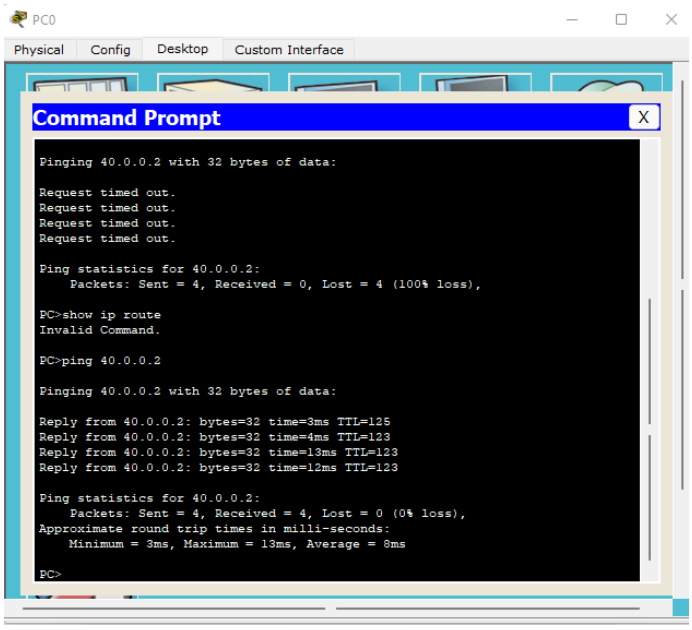


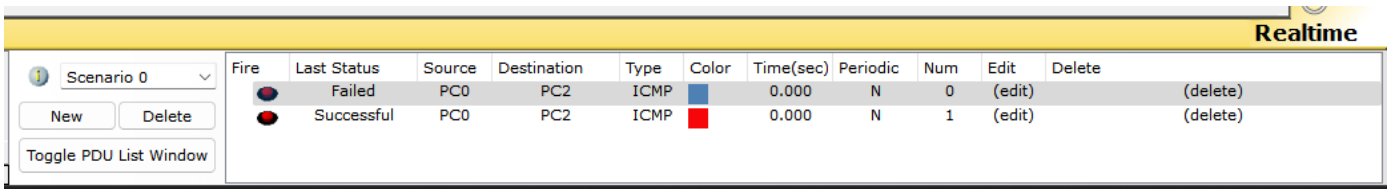
Screen shots/ output:

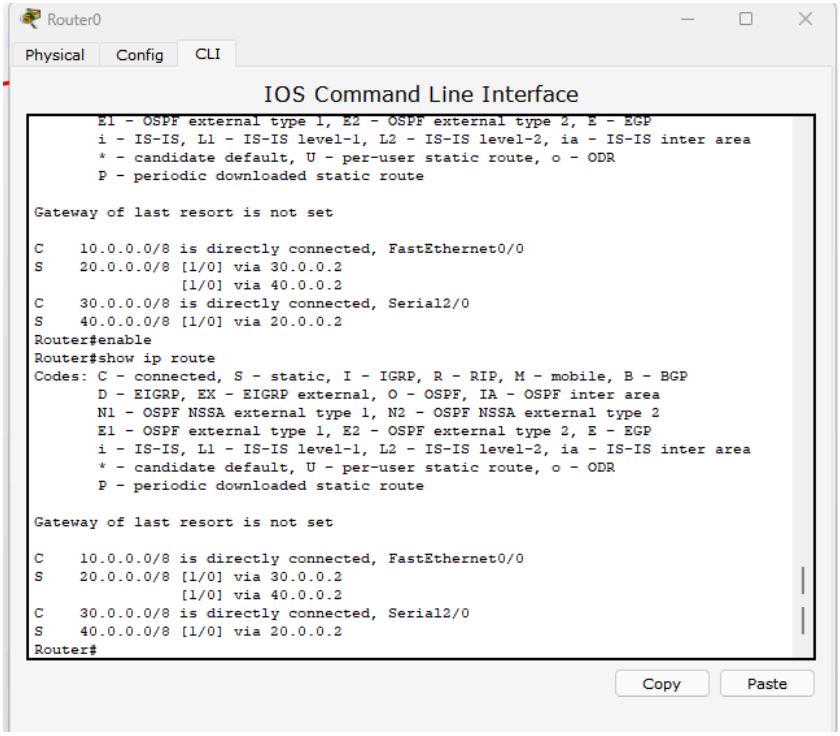




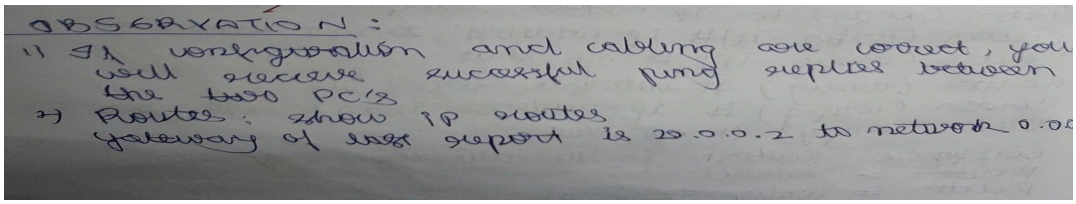

PC1
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.1
Pinging 10.0.0.1 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 10.0.0.1:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
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=9ms TTL=125
Reply from 10.0.0.1: bytes=32 time=13ms TTL=125
Reply from 10.0.0.1: bytes=32 time=10ms TTL=125
Reply from 10.0.0.1: bytes=32 time=15ms 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 = 9ms, Maximum = 15ms, Average = 11ms
PC>

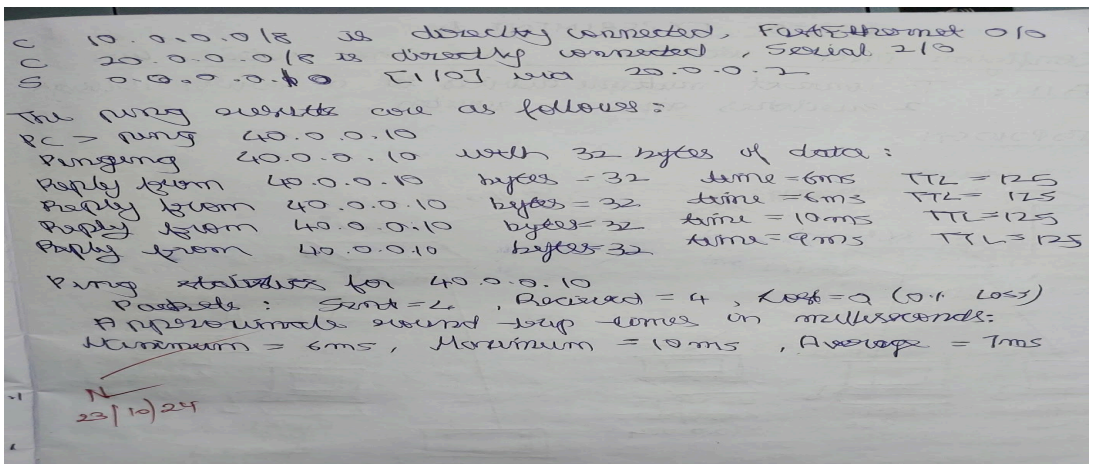

PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Pinging 40.0.0.2 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 40.0.0.2:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>show ip route
Invalid Command.
PC>ping 40.0.0.2
Pinging 40.0.0.2 with 32 bytes of data:
Reply from 40.0.0.2: bytes=32 time=3ms TTL=125
Reply from 40.0.0.2: bytes=32 time=4ms TTL=123
Reply from 40.0.0.2: bytes=32 time=13ms TTL=123
Reply from 40.0.0.2: bytes=32 time=12ms TTL=123
Ping statistics for 40.0.0.2:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 3ms, Maximum = 13ms, Average = 8ms
PC>


Realtime
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Type
Color
Time(sec)
Periodic
Num
Edit
Delete
Failed PC0 PC2 ICMP 0.000 N 0 (edit) (delete)
Successful PC0 PC2 ICMP 0.000 N 1 (edit) (delete)



Observation:

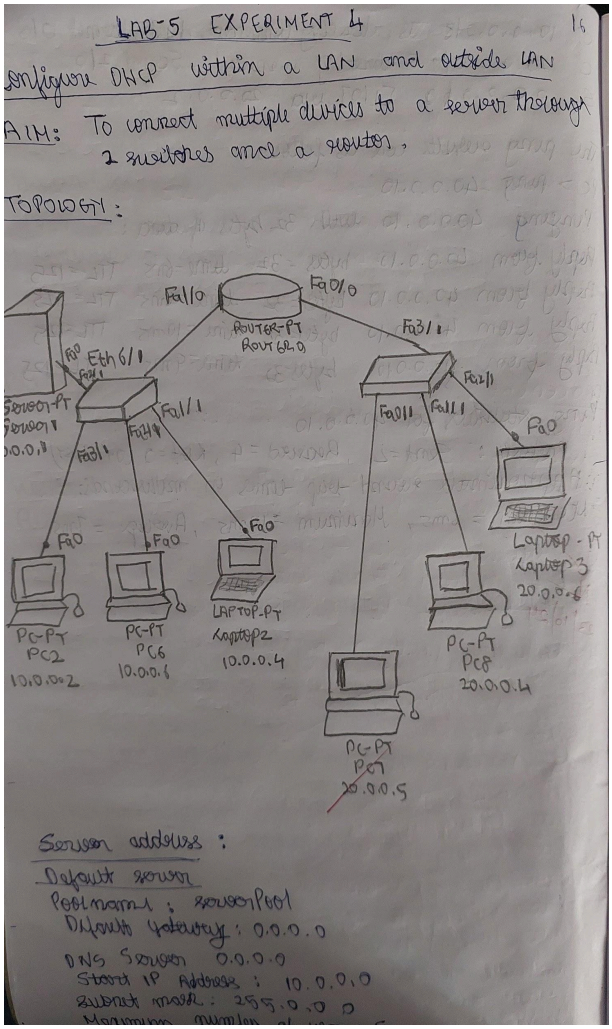


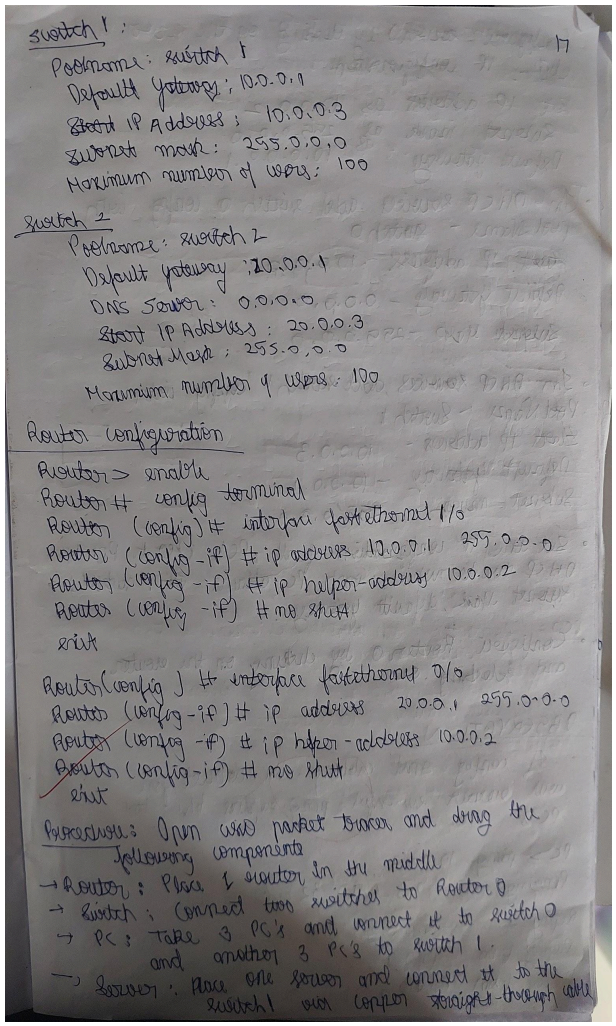


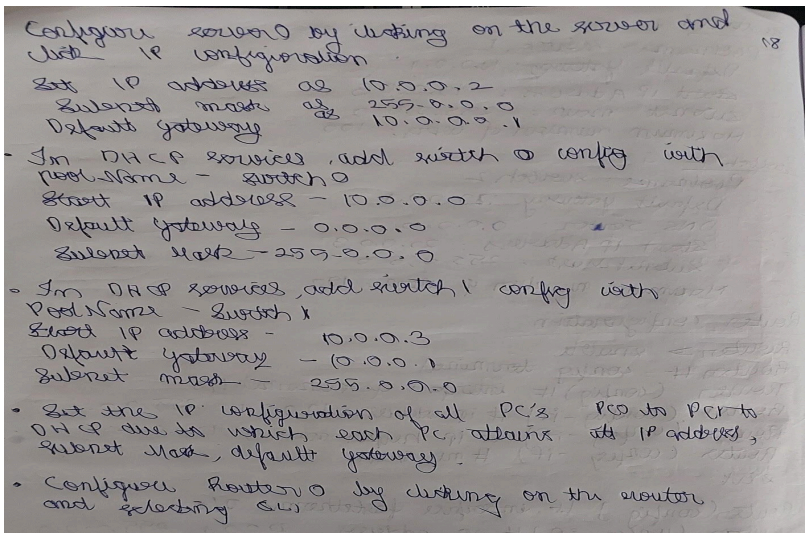
Program 4

Aim of the program:
Configure IP address of the host using DHCP server within and outside a LAN.

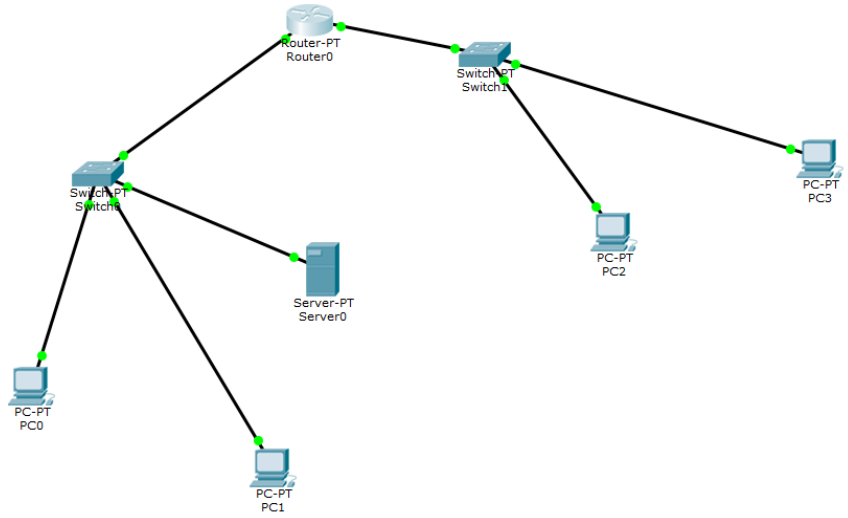
Procedure along with the topology:

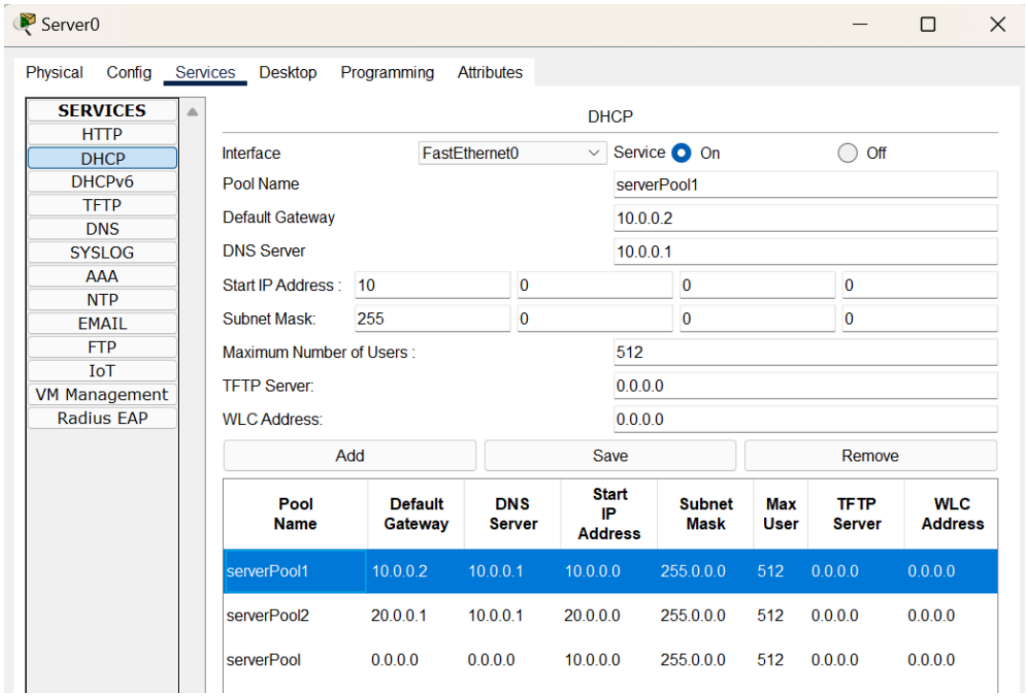




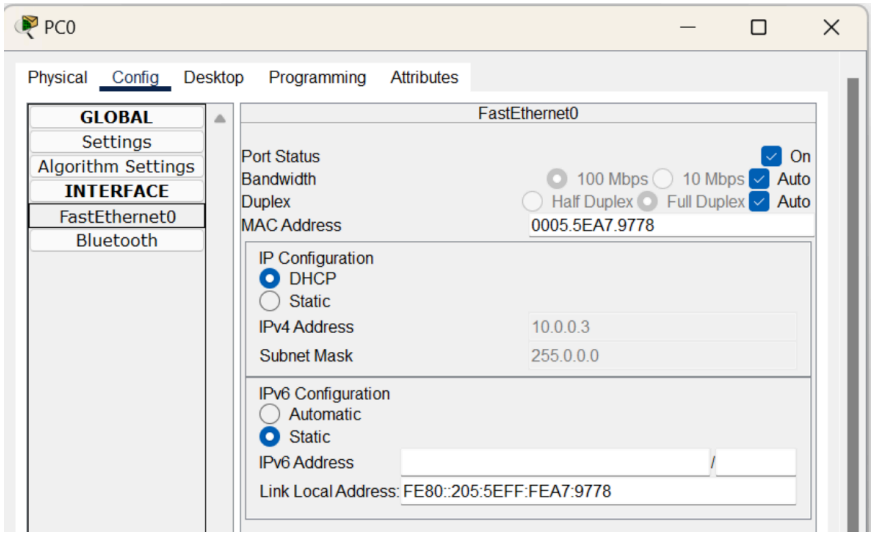


Screen shots/ output:

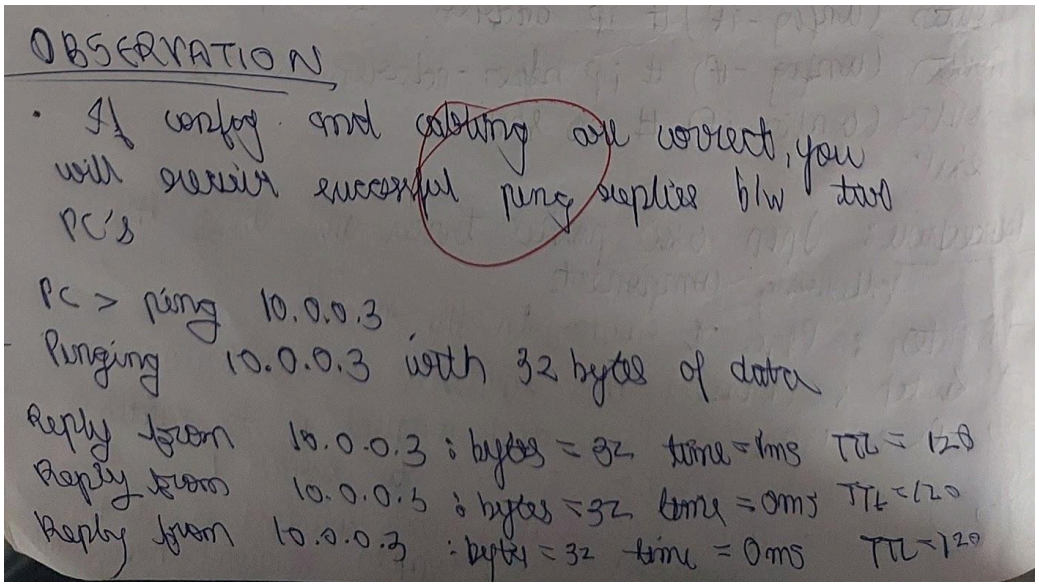




Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	Laptop0	ICMP		0.000	N	0	(edit)	
	Successful	PC1	Laptop1	ICMP		0.004	N	1	(edit)	



Observation:



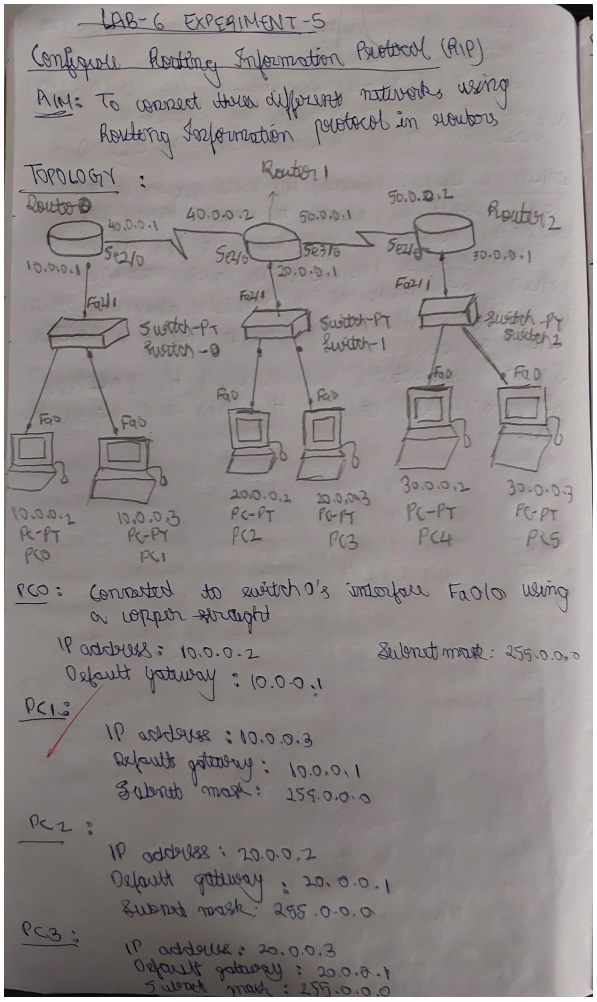
Reply from 10.0.0.3 : bytes = 32 time = 2ms TTL = 128
Ping statistics for 10.0.0.3:
Packets : Sent = 4, Received = 4, Loss = 0 (0% Loss)
Approximate round trip times in milliseconds:
Minimum = 0ms, Maximum = 2ms, Average = 0ms

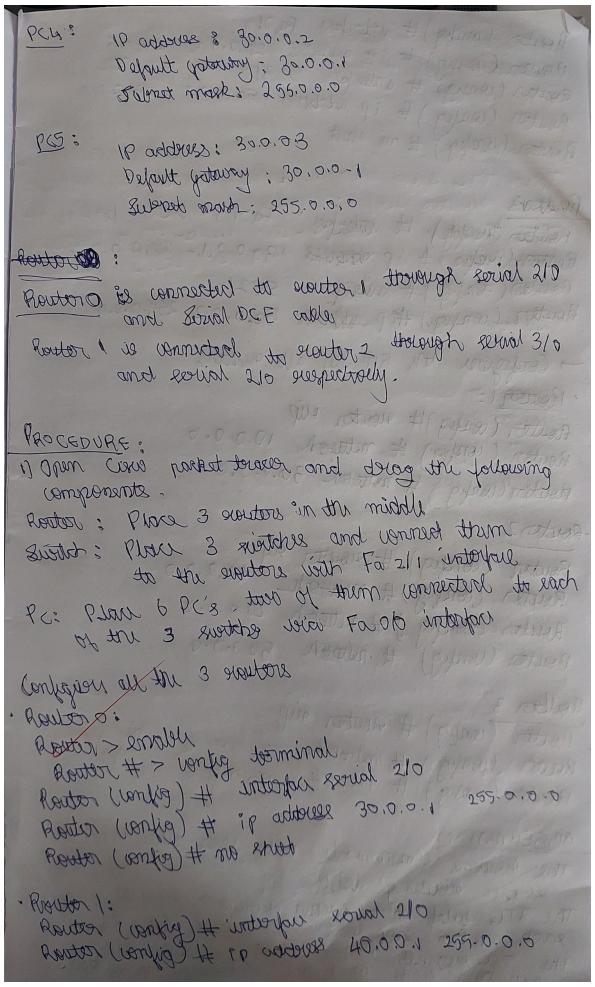
Within a LAN : Placing the DHCP server in the same subnet as clients to ensure broadcasts reach the server directly. Dynamic ip's are given to the systems connected in same network.
When we have to dynamically assign ip address to another network we do it using a router and a server.
If the connections are successful the ip addresses are assigned within the LAN and outside the LAN

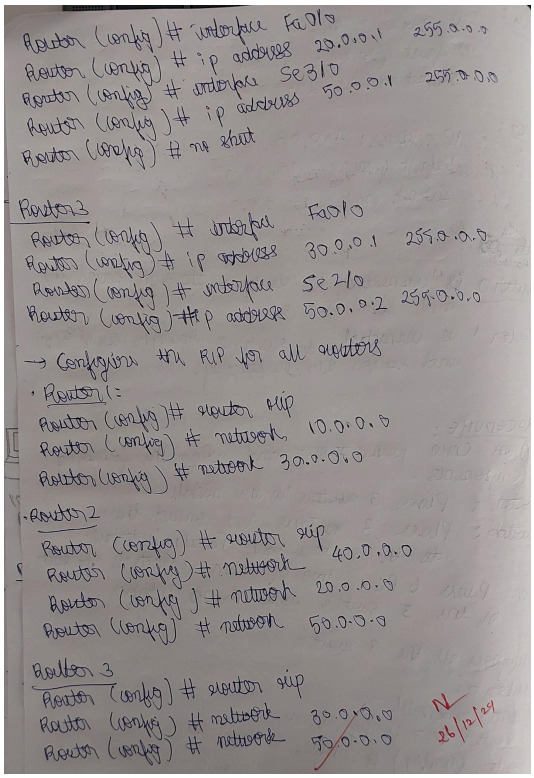
Program 5

Aim of the program:
Configure RIP routing Protocol in Routers

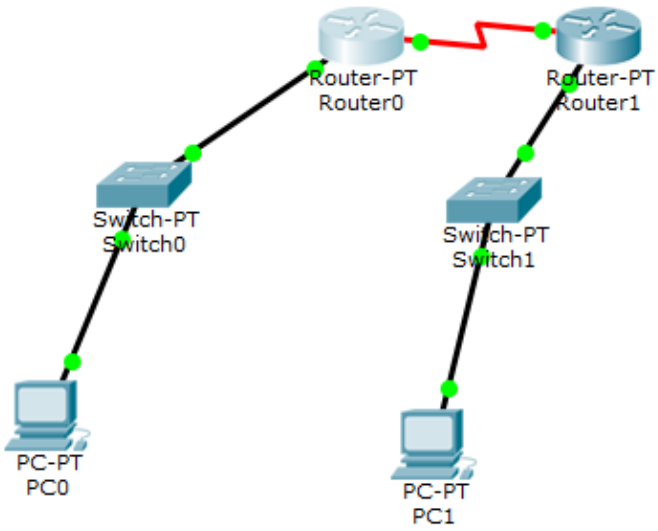
Procedure along with the topology:

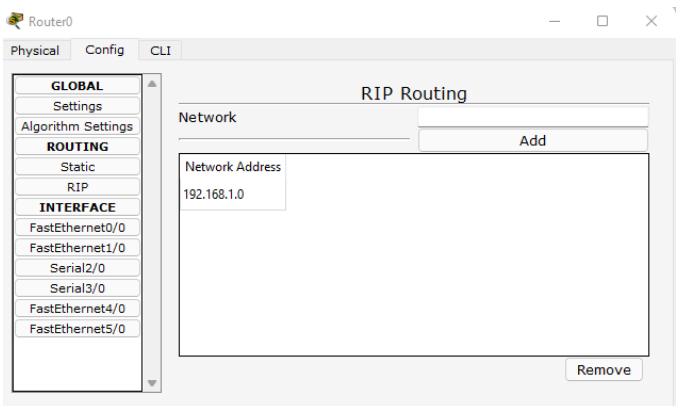


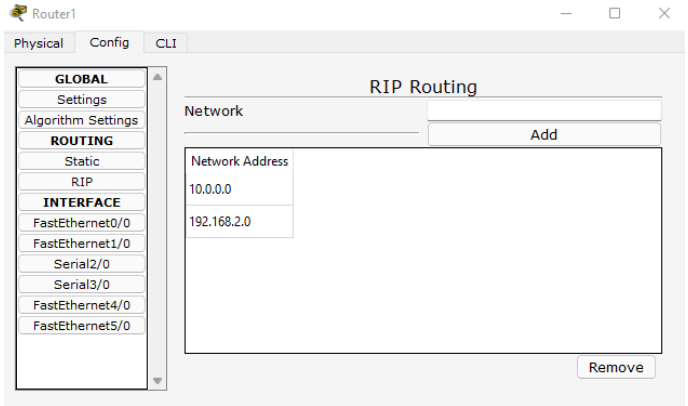




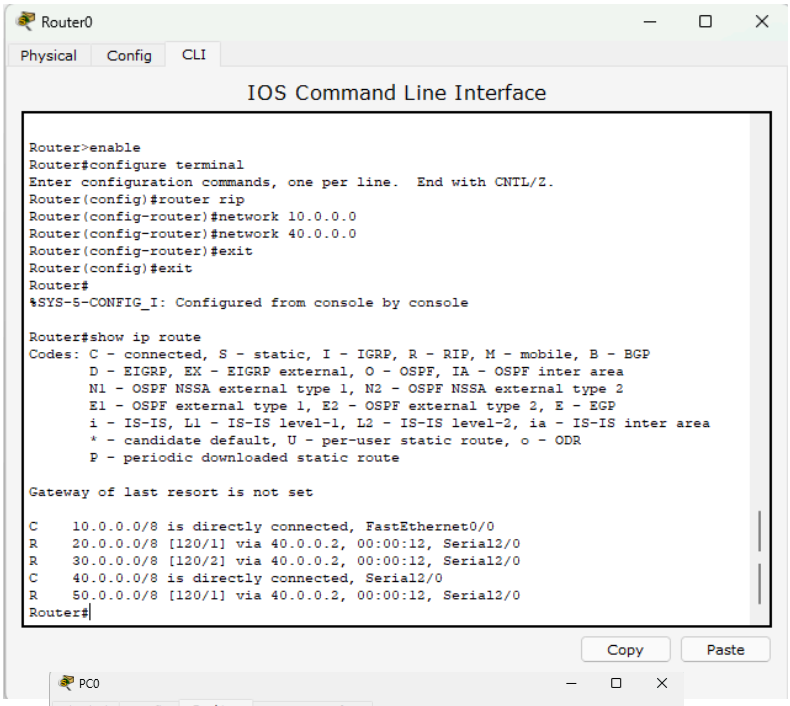
Screen shots/ output:

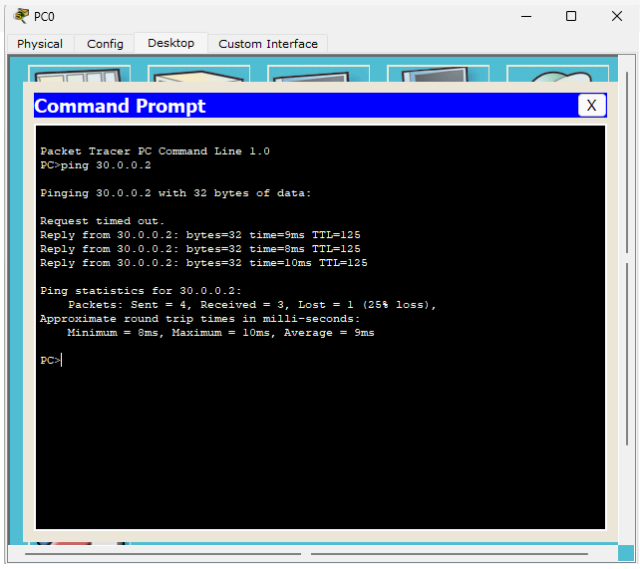




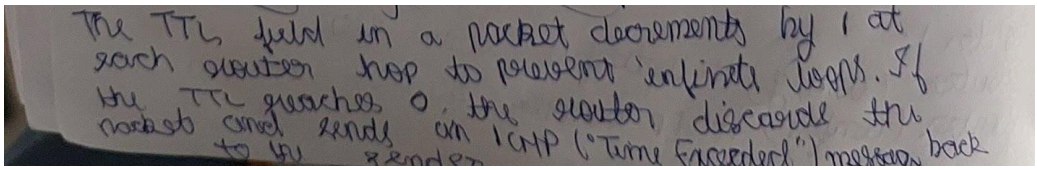


Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Failed	PC0	PC1	ICMP		0.000	N	0	(edit)	(delete)
	Successful	Router0	Router1	ICMP		0.000	N	1	(edit)	(delete)
	Successful	PC0	PC1	ICMP		0.000	N	2	(edit)	(delete)





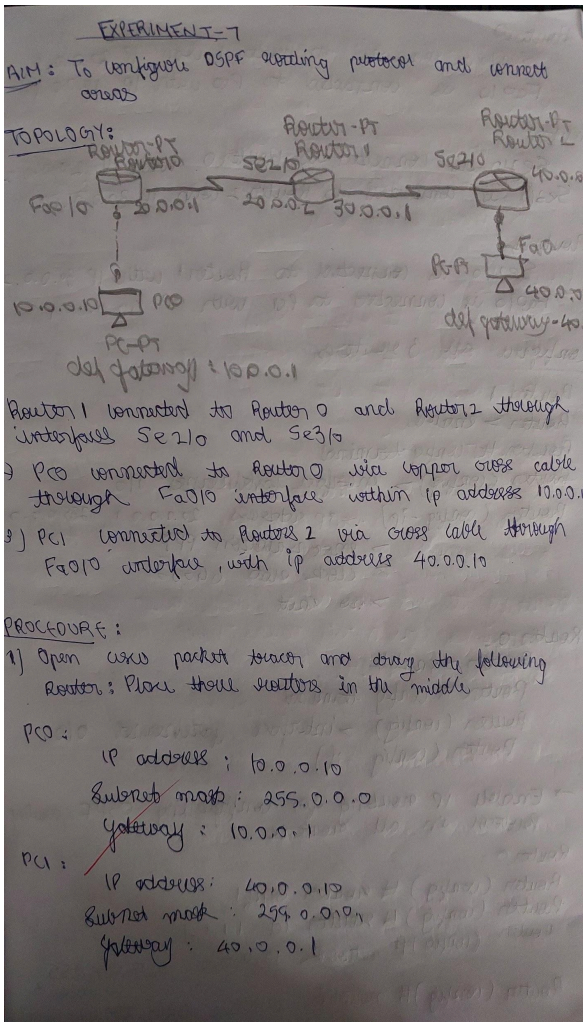
Observation:

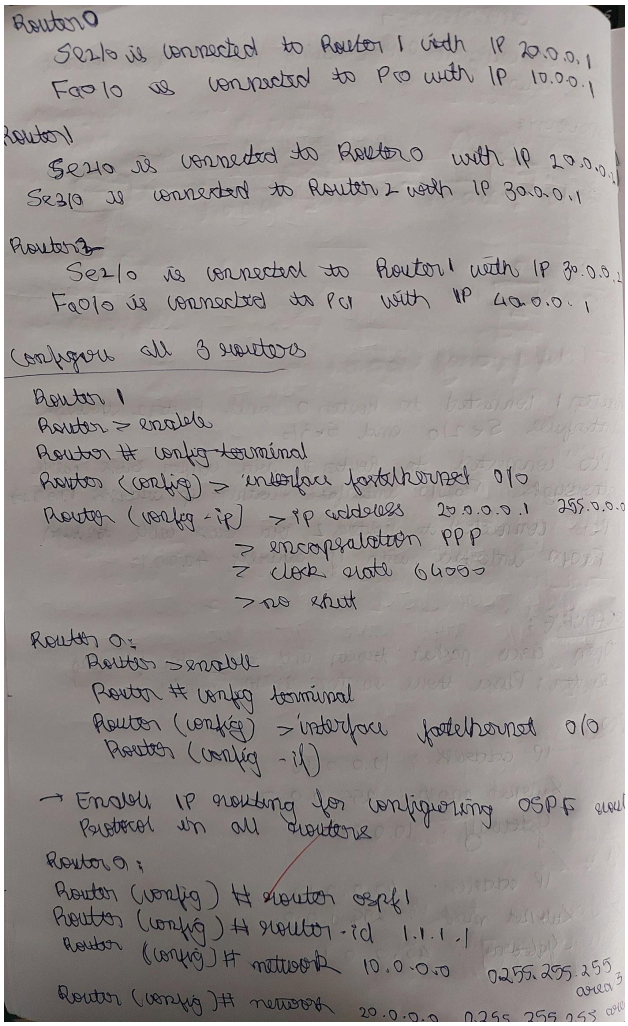


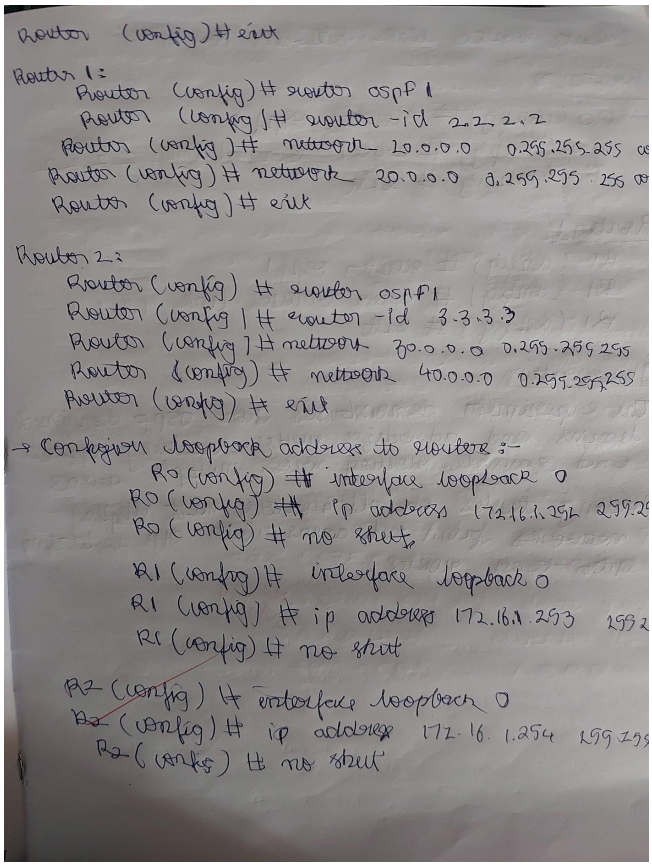
Program 6

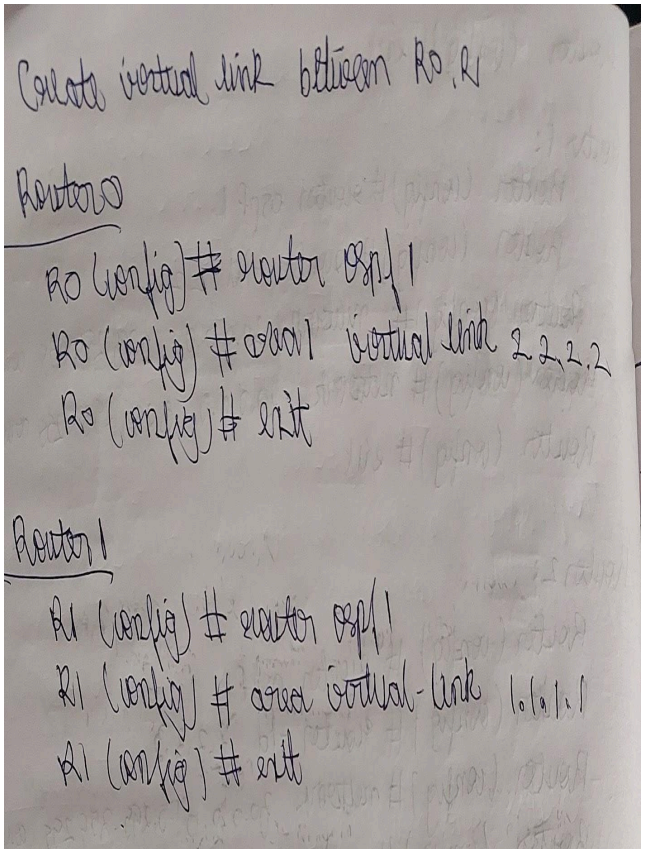
Configure OSPF routing protocol

Procedure along with the topology

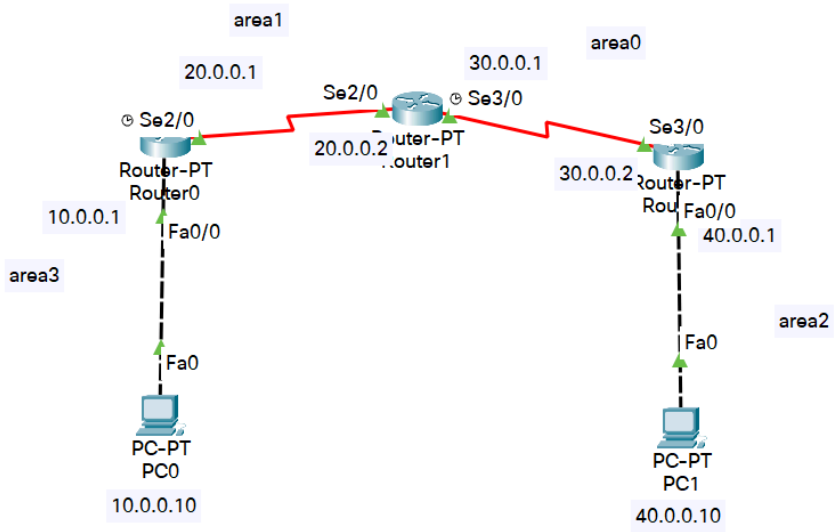








Screen shots/ output



Router0

```
Router>enable
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface Fa0/0
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed
state to up

Router(config-if)#exit
Router(config)#interface Se2/0
Router(config-if)#ip address 20.0.0.1 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial2/0, changed state to down
Router(config-if)#exit
Router(config)#
```

Router1

```
Router>enable
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface Se2/0
Router(config-if)#ip address 20.0.0.2 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial2/0, changed state to
up

Router(config-if)#exit
Router(config)#interface Se3/0
Router(config-if)#ip address 30.0.0.1 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#clock rate 64000
Router(config-if)#no dhutdown
                   ^
% Invalid input detected at '^' marker.

Router(config-if)#no shutdown
```

Router2

```
Router>enable
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface Fa0/0
Router(config-if)#ip address 40.0.0.1 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet0/0, changed
state to up
%IP-4-DUPADDR: Duplicate address 40.0.0.1 on FastEthernet0/0, sourced by
000D.BDDA.0123

Router(config-if)#exit
Router(config)#interface Se3/0
Router(config-if)#ip address 30.0.0.2 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface Serial3/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial3/0, changed state to
up
```

OSPF Routing Protocol

Router0

```
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#router-id 1.1.1.1
Router(config-router)#network 10.0.0.0 0.255.255.255 area 3
Router(config-router)#network 20.0.0.0 0.255.255.255 area 1
Router(config-router)#end
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#sho
00:27:19: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on Serial2/0 from LOADING to FULL, Loading Done
w ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
     20.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       20.0.0.0/8 is directly connected, Serial2/0
C       20.0.0.2/32 is directly connected, Serial2/0
O IA 30.0.0.0/8 [110/128] via 20.0.0.2, 00:00:02, Serial2/0
O IA 40.0.0.0/8 [110/129] via 20.0.0.2, 00:00:02, Serial2/0
```

Router1

```
Router>enable
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#router-id 2.2.2.2
Router(config-router)#network 20.0.0.0 0.255.255.255 area 1
Router(config-router)#network 30.0.0.0 0.255.255.255 area 0
Router(config-router)#end
Router#
%SYS-5-CONFIG_I: Configured from console by console

00:26:21: %OSPF-5-ADJCHG: Process 1, Nbr 3.3.3.3 on Serial3/0 from LOADING to FULL, Loading Done

00:27:18: %OSPF-5-ADJCHG: Process 1, Nbr 1.1.1.1 on Serial2/0 from LOADING to FULL, Loading Done

Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

     20.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       20.0.0.0/8 is directly connected, Serial2/0
C       20.0.0.1/32 is directly connected, Serial2/0
     30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       30.0.0.0/8 is directly connected, Serial3/0
C       30.0.0.2/32 is directly connected, Serial3/0
O IA 40.0.0.0/8 [110/65] via 30.0.0.2, 00:02:00, Serial3/0
```

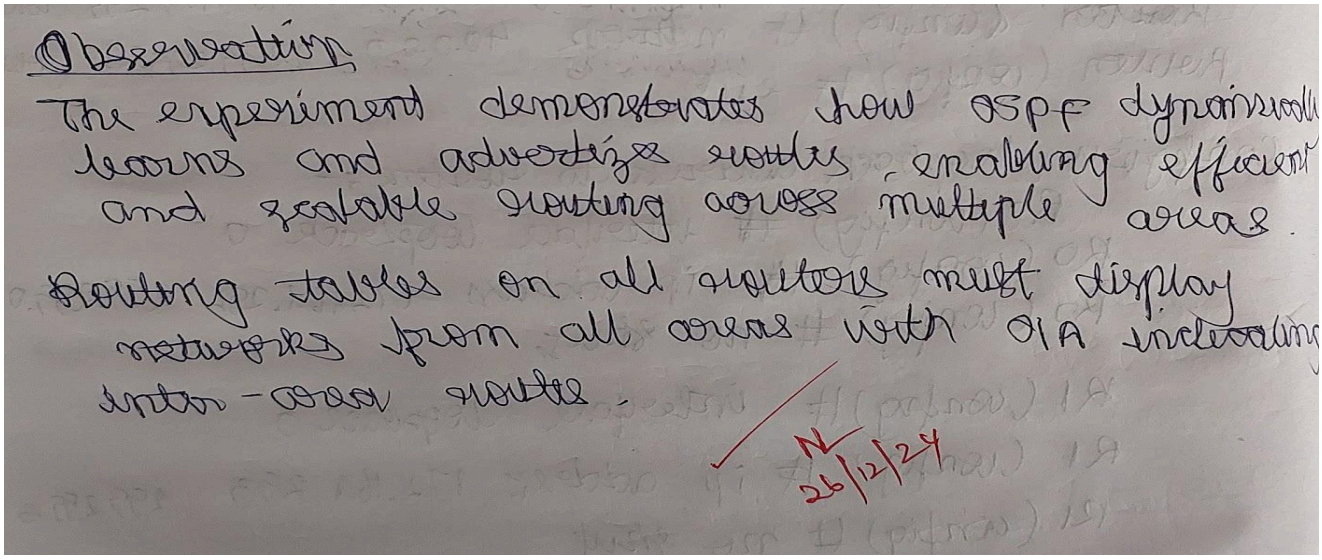
Router2

```
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router ospf 1
Router(config-router)#router-id 3.3.3.3
Router(config-router)#network 40.0.0.0 0.255.255.255 area 2
Router(config-router)#network 30.0.0.0 0.255.255.255 area 0
Router(config-router)#end
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#
00:26:19: %OSPF-5-ADJCHG: Process 1, Nbr 2.2.2.2 on Serial3/0 from LOADING to FULL, Loading Done

Router#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, ia - IS-IS inter area
       * - candidate default, U - per-user static route, o - ODR
       P - periodic downloaded static route

Gateway of last resort is not set

O IA 20.0.0.0/8 [110/128] via 30.0.0.1, 00:02:45, Serial3/0
     30.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C       30.0.0.0/8 is directly connected, Serial3/0
C       30.0.0.1/32 is directly connected, Serial3/0
C    40.0.0.0/8 is directly connected, FastEthernet0/0
```

Pinging

```
C:\>ping 40.0.0.10

Pinging 40.0.0.10 with 32 bytes of data:

Reply from 40.0.0.10: bytes=32 time=24ms TTL=125
Reply from 40.0.0.10: bytes=32 time=18ms TTL=125
Reply from 40.0.0.10: bytes=32 time=18ms TTL=125
Reply from 40.0.0.10: bytes=32 time=20ms 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 = 18ms, Maximum = 24ms, Average = 20ms
```

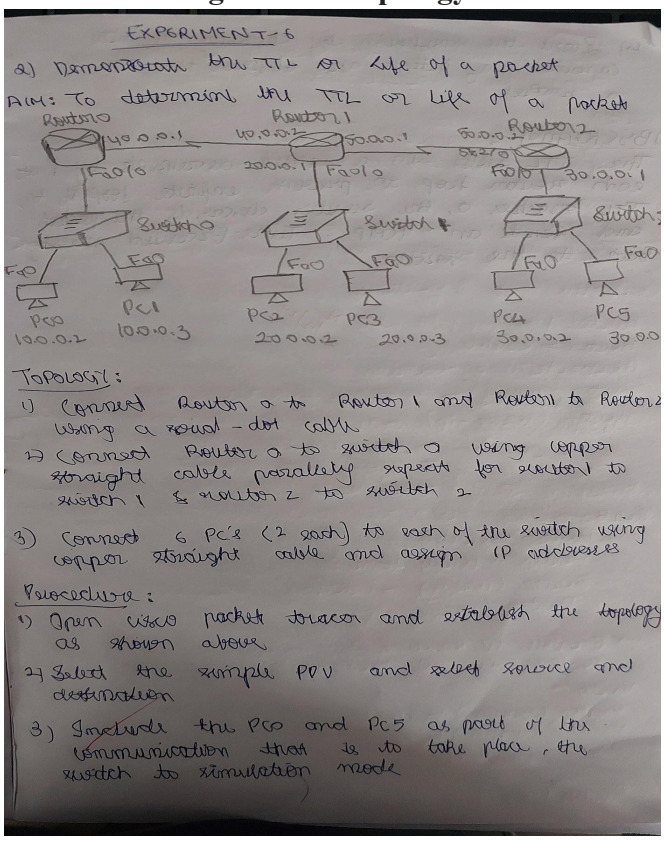
Observation



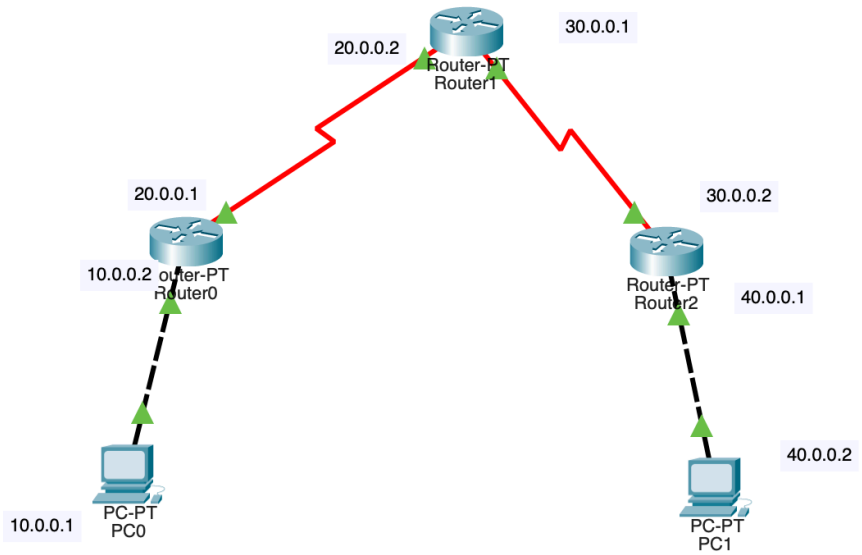
Program 7

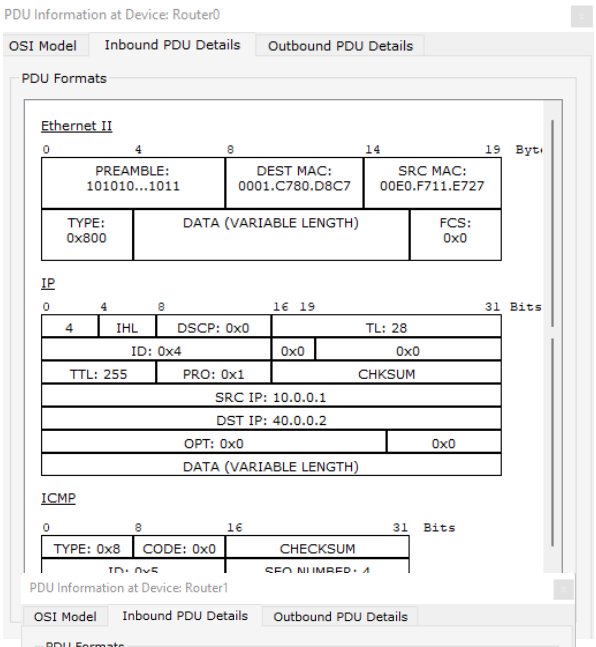
Aim of the program:
Demonstrate the TTL/ Life of a Packet

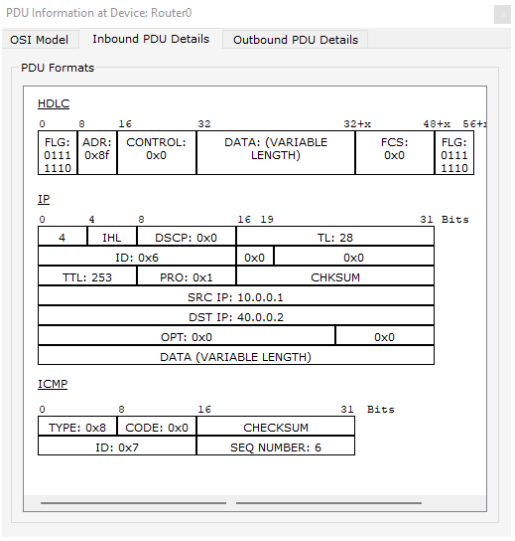
Procedure along with the topology:

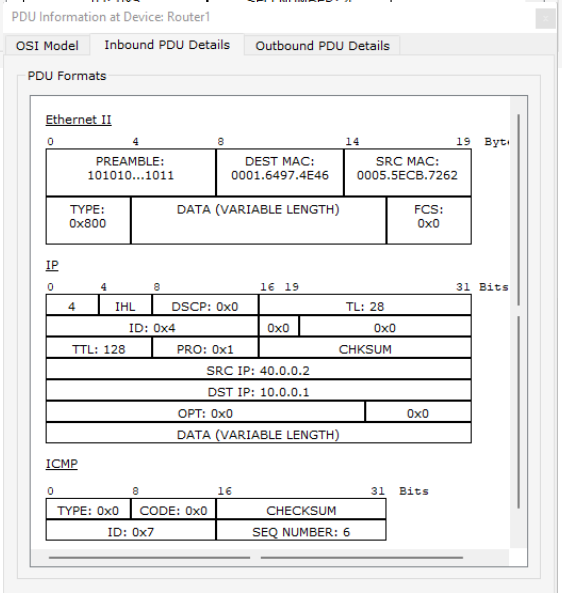


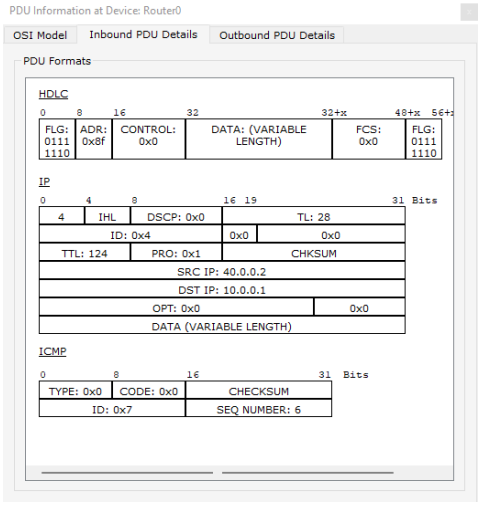
Screen shots/ output:



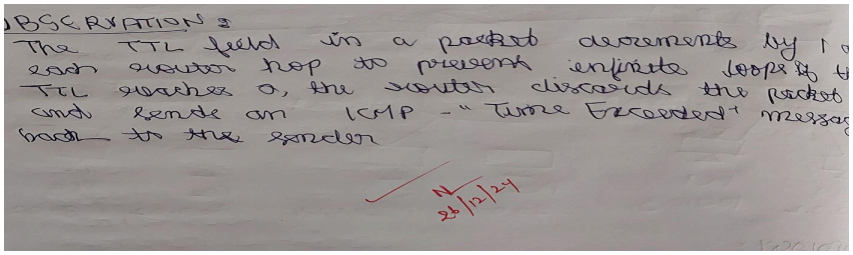








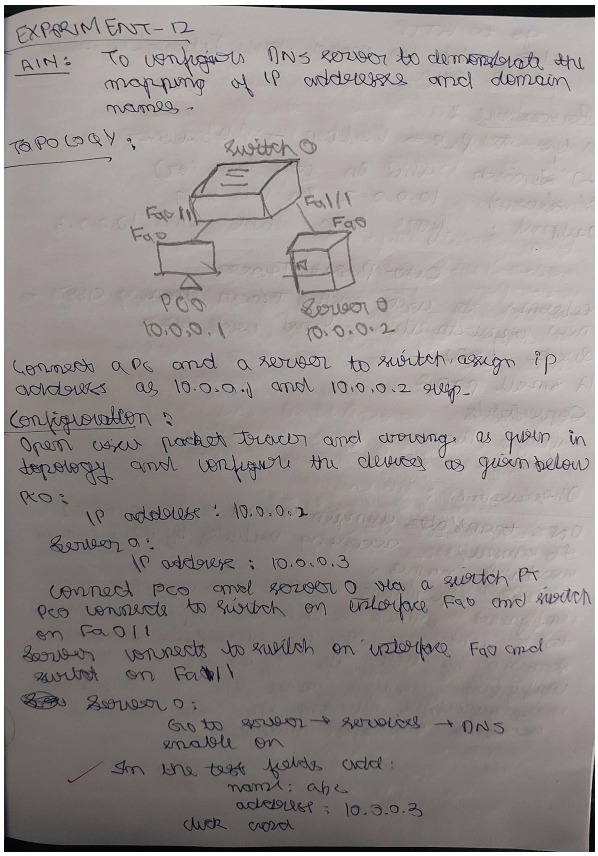
Observation:

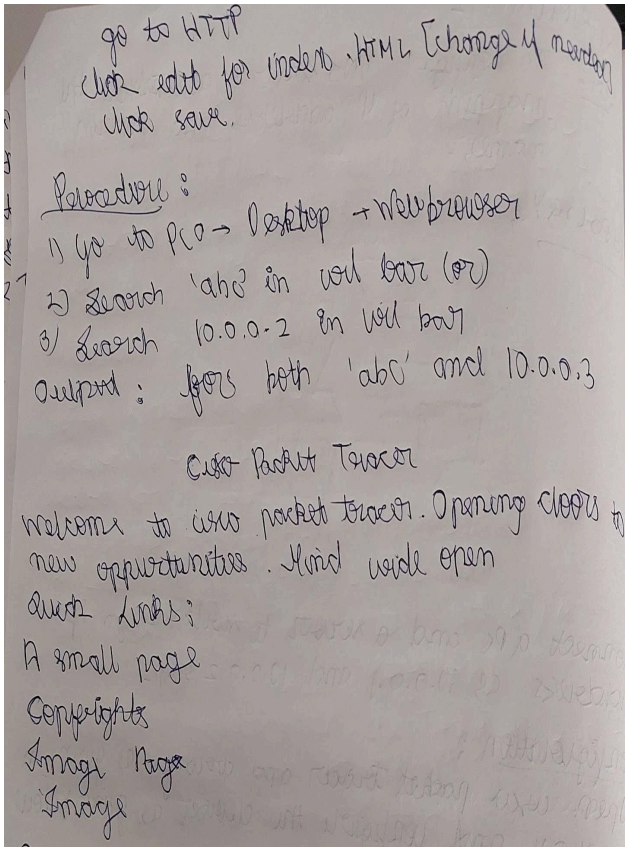


Program 8

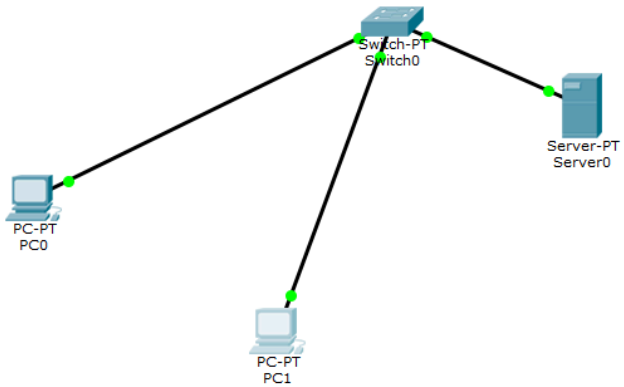
Aim of the program:
Configure Web Server, DNS within a LAN.

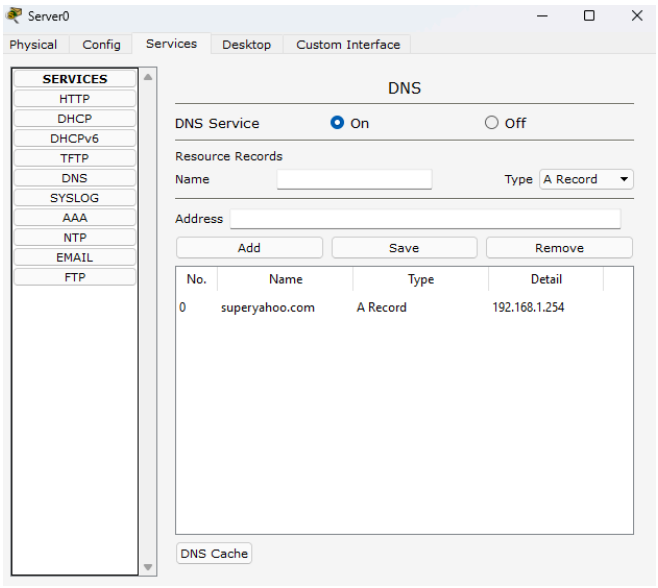
Procedure along with the topology:

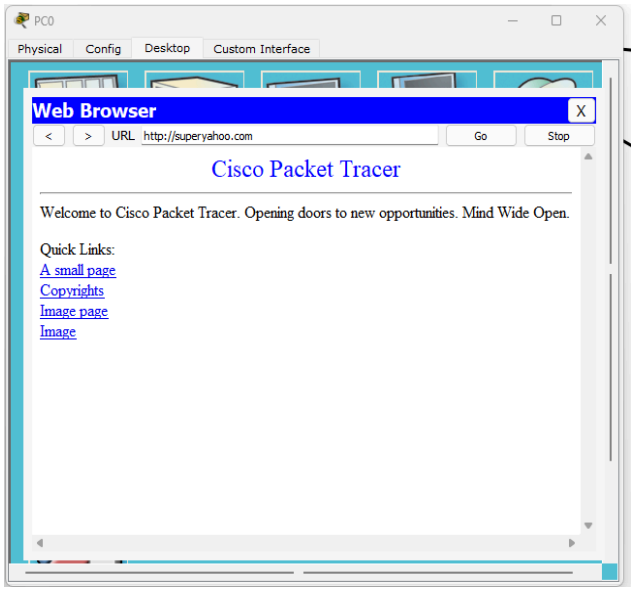


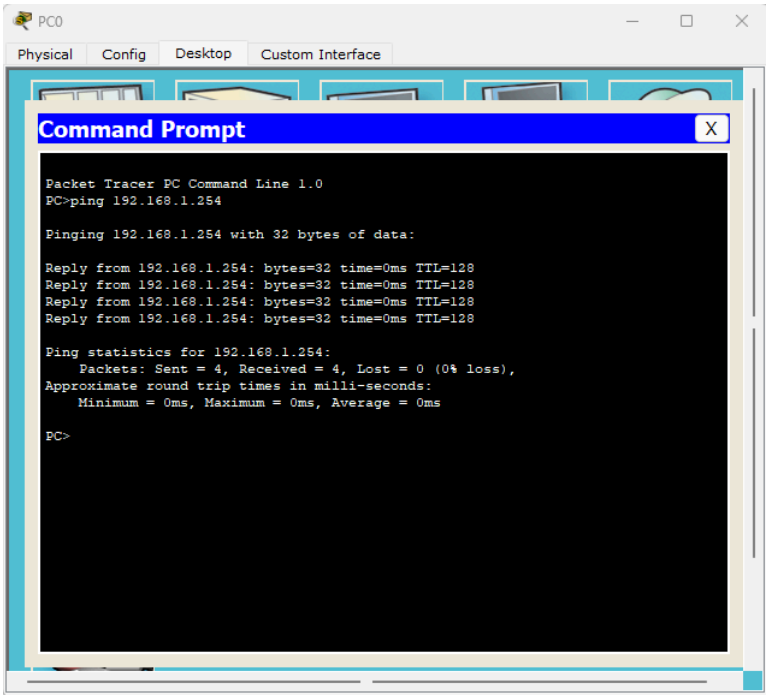


Screen shots/ output:

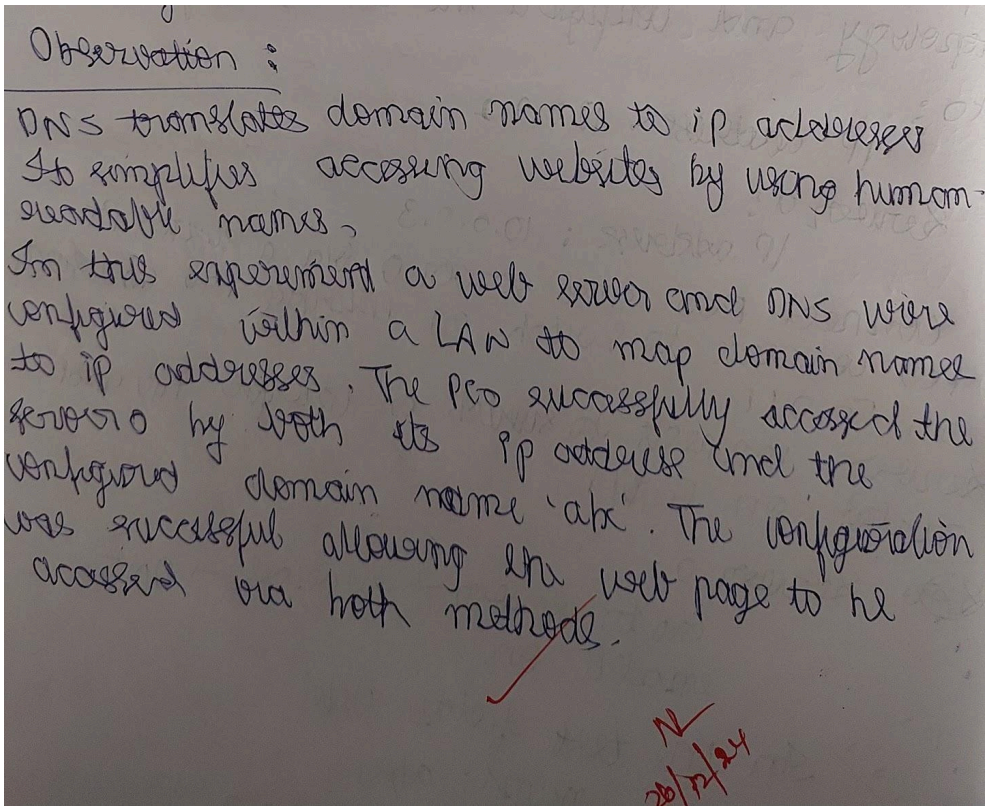



Server0
Physical
Config
Services
Desktop
Custom Interface
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
DNS
DNS Service
On
Off
Resource Records
Name
Type
A Record
Address
Add
Save
Remove
No.
Name
Type
Detail
0
superyahoo.com
A Record
192.168.1.254
DNS Cache


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


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
Packet Tracer PC Command Line 1.0
PC>ping 192.168.1.254
Pinging 192.168.1.254 with 32 bytes of data:
Reply from 192.168.1.254: bytes=32 time=0ms TTL=128
Reply from 192.168.1.254: bytes=32 time=0ms TTL=128
Reply from 192.168.1.254: bytes=32 time=0ms TTL=128
Reply from 192.168.1.254: bytes=32 time=0ms TTL=128
Ping statistics for 192.168.1.254:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
PC>

Observation:



Observation :

DNS translates domain names to ip addresses. It simplifies accessing websites by using human-readable names.

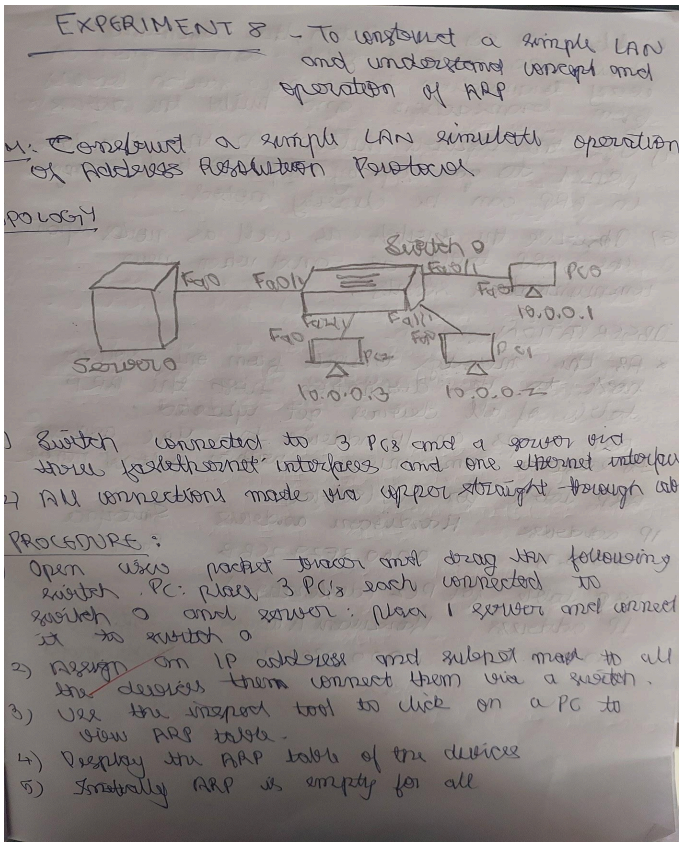
In this experiment a web server and DNS were configured within a LAN to map domain names to ip addresses. The PC successfully accessed the server by both its ip address and the configured domain name 'abc'. The configuration was successful allowing the web page to be accessed via both methods.

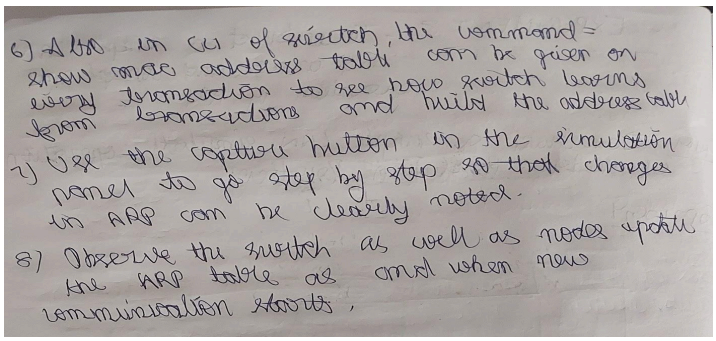
26/12/24

Program 9

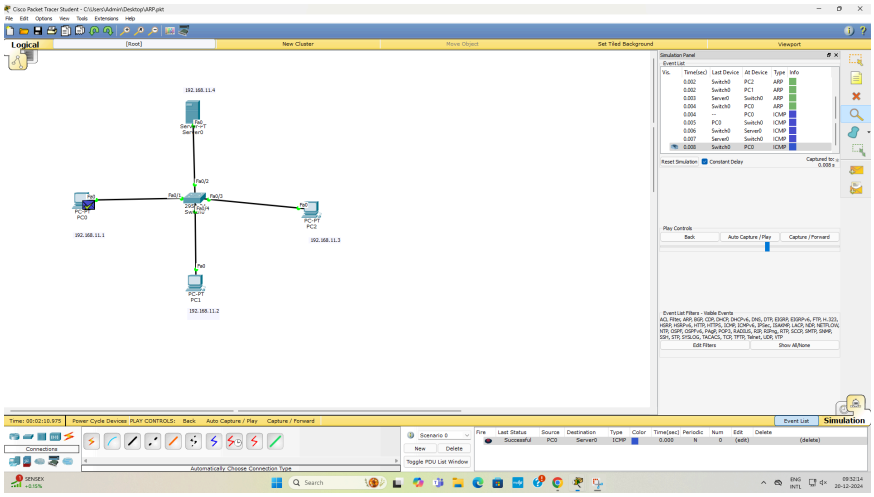
Aim of the program:
To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)

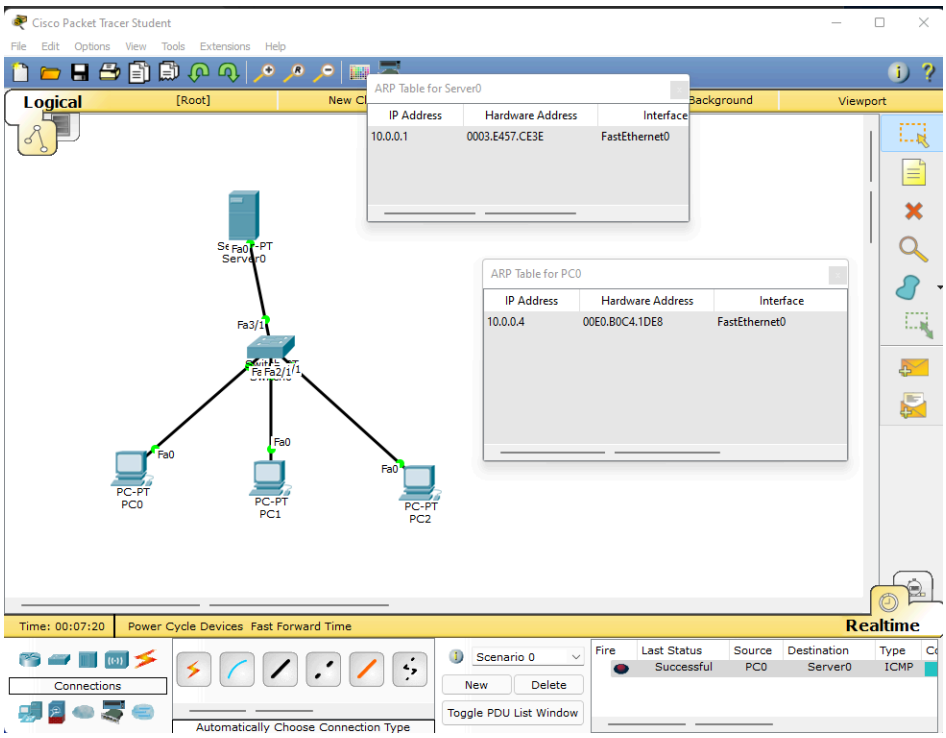
Procedure along with the topology:



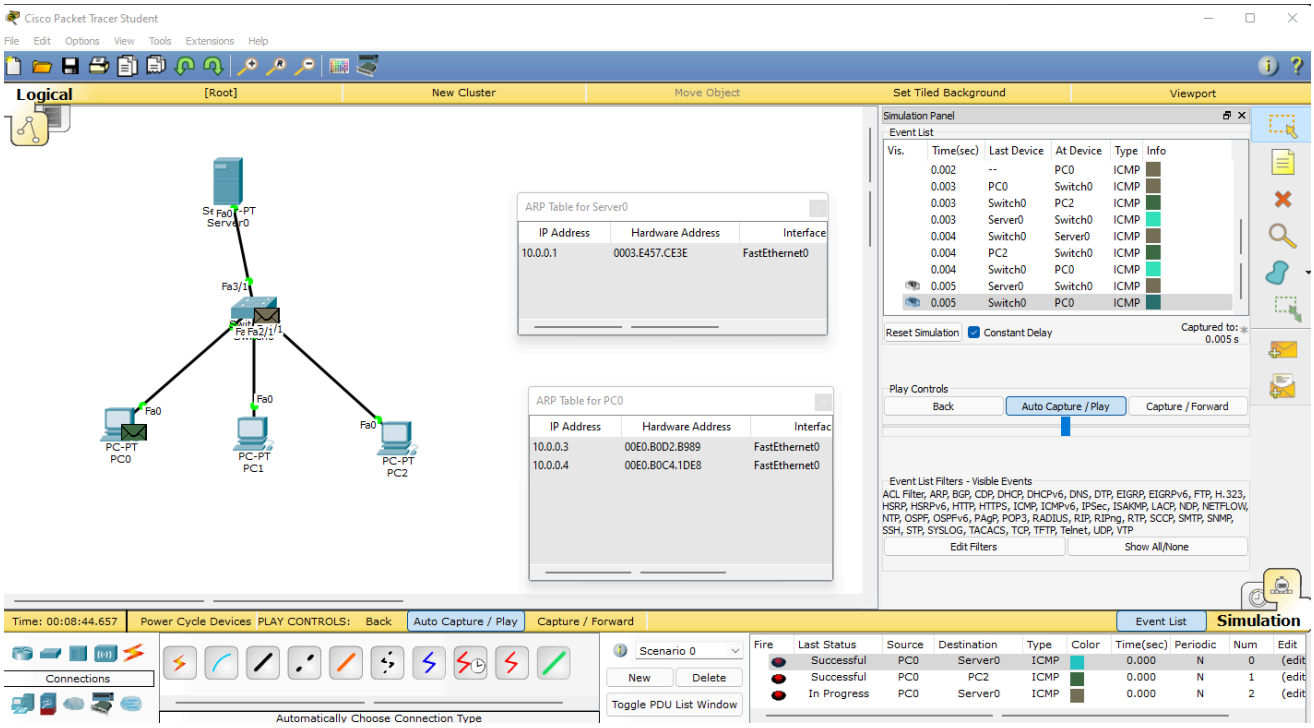


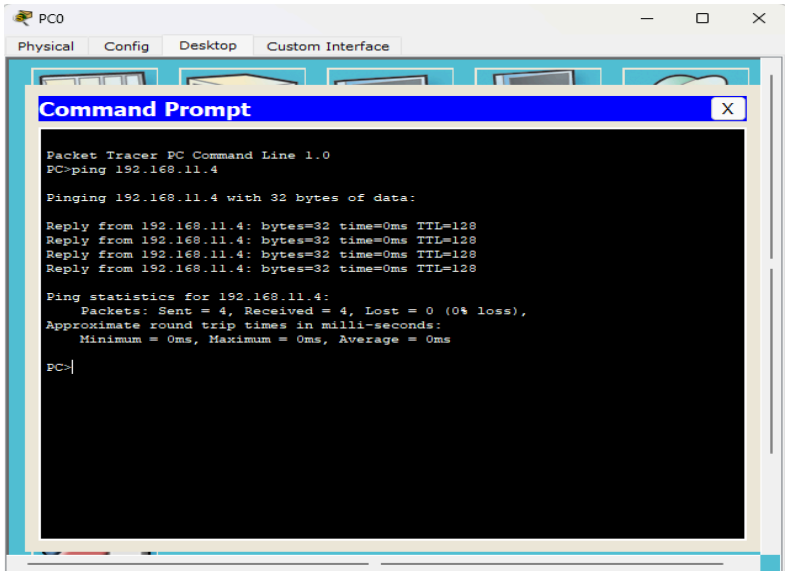
Screen shots/ output:

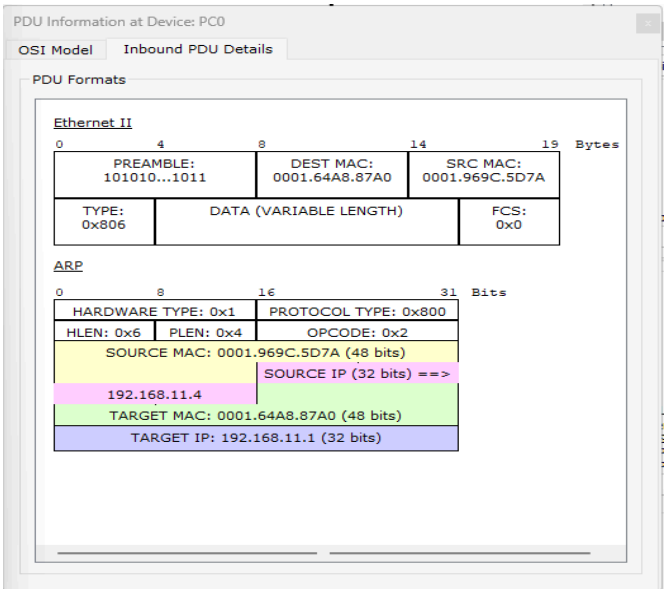




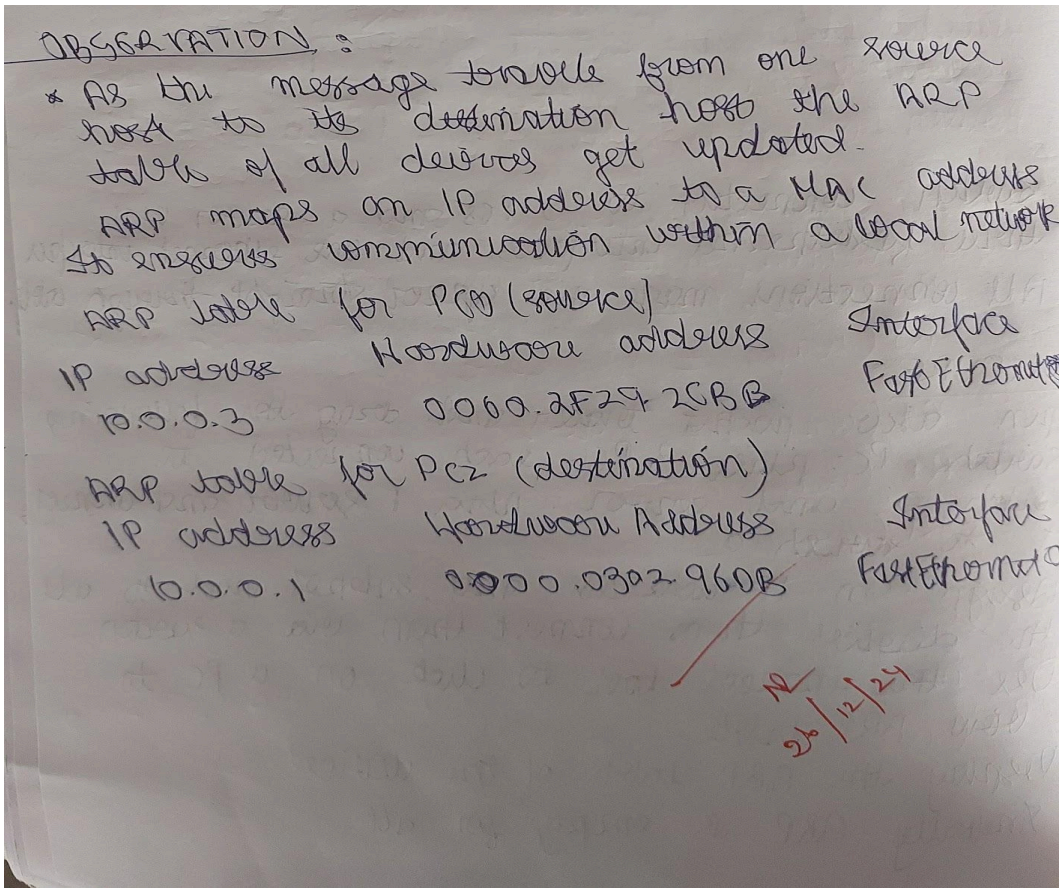
Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit
	Successful	PC0	Server0	ICMP		0.000	N	0	(edit
	Successful	PC0	PC2	ICMP		0.000	N	1	(edit
	In Progress	PC0	Server0	ICMP		0.000	N	2	(edit


Cisco Packet Tracer Student
File Edit Options View Tools Extensions Help
Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
Server0
Fa0
Fa3/1
Fa2/1/1
PC-PT PC0
PC-PT PC1
PC-PT PC2
ARP Table for Server0
IP Address
Hardware Address
Interface
10.0.0.1
0003.E457.CE3E
FastEthernet0
ARP Table for PC0
10.0.0.3
00E0.B0D2.B989
10.0.0.4
00E0.B0C4.1DE8
Simulation Panel
Event List
Vis.
Time(sec)
Last Device
At Device
Type
Info
Reset Simulation
Constant Delay
Play Controls
Back
Auto Capture / Play
Capture / Forward
Event List Filters - Visible Events
Edit Filters
Show All/None
Time: 00:08:44.657
Power Cycle Devices
PLAY CONTROLS:
Simulation
Connections
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window
Fire
Last Status
Source
Destination
Color
Periodic
Num
Edit
Successful
In Progress


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
Packet Tracer PC Command Line 1.0
PC>ping 192.168.11.4
Pinging 192.168.11.4 with 32 bytes of data:
Reply from 192.168.11.4: bytes=32 time=0ms TTL=128
Reply from 192.168.11.4: bytes=32 time=0ms TTL=128
Reply from 192.168.11.4: bytes=32 time=0ms TTL=128
Reply from 192.168.11.4: bytes=32 time=0ms TTL=128
Ping statistics for 192.168.11.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
PC>


PDU Information at Device: PC0
OSI Model
Inbound PDU Details
PDU Formats
Ethernet II
0
4
8
14
19
Bytes
PREAMBLE: 101010...1011
DEST MAC: 0001.64A8.87A0
SRC MAC: 0001.969C.5D7A
TYPE: 0x806
DATA (VARIABLE LENGTH)
FCS: 0x0
ARP
0
8
16
31
Bits
HARDWARE TYPE: 0x1
PROTOCOL TYPE: 0x800
HLEN: 0x6
PLEN: 0x4
OPCODE: 0x2
SOURCE MAC: 0001.969C.5D7A (48 bits)
SOURCE IP (32 bits) ==>
192.168.11.4
TARGET MAC: 0001.64A8.87A0 (48 bits)
TARGET IP: 192.168.11.1 (32 bits)

Observation:

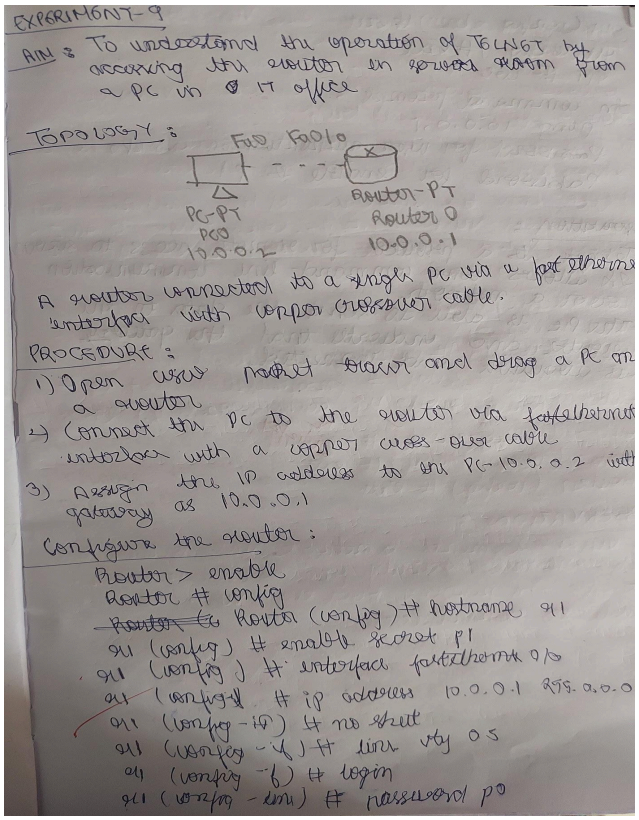


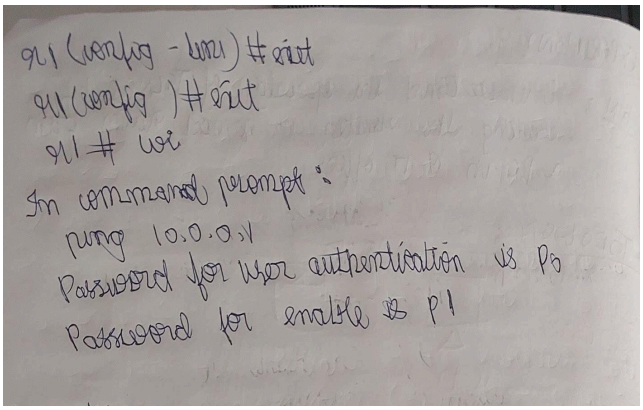
Program 10

Aim of the program:

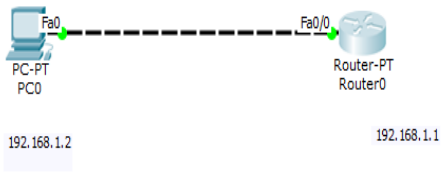
To understand the operation of TELNET by accessing the router in the server room from a PC in the IT office.

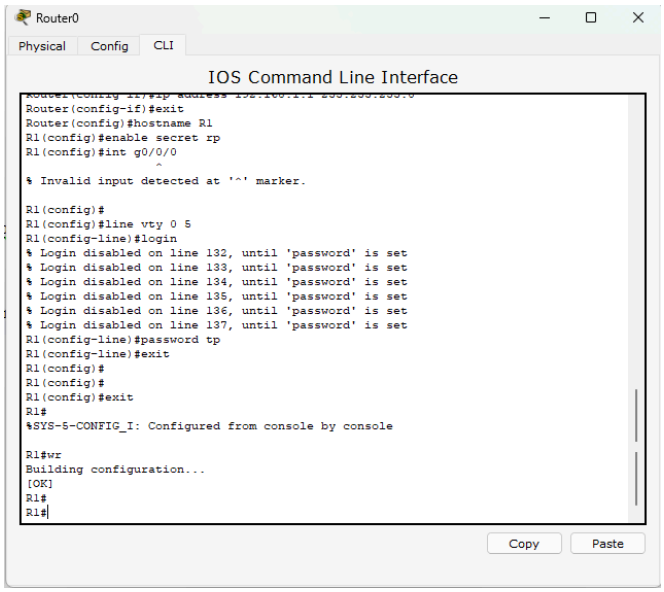
Procedure along with the topology:

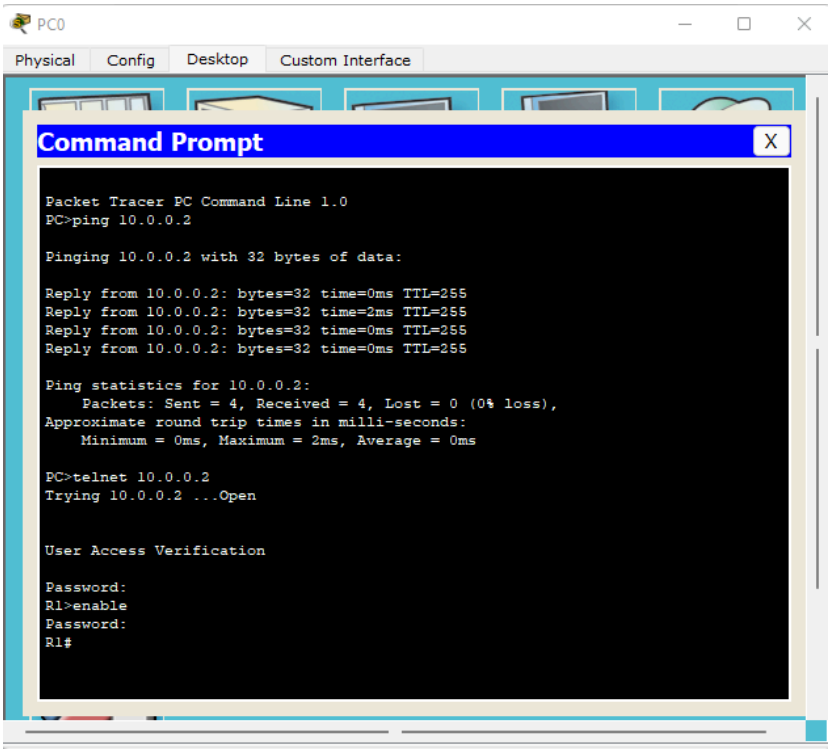




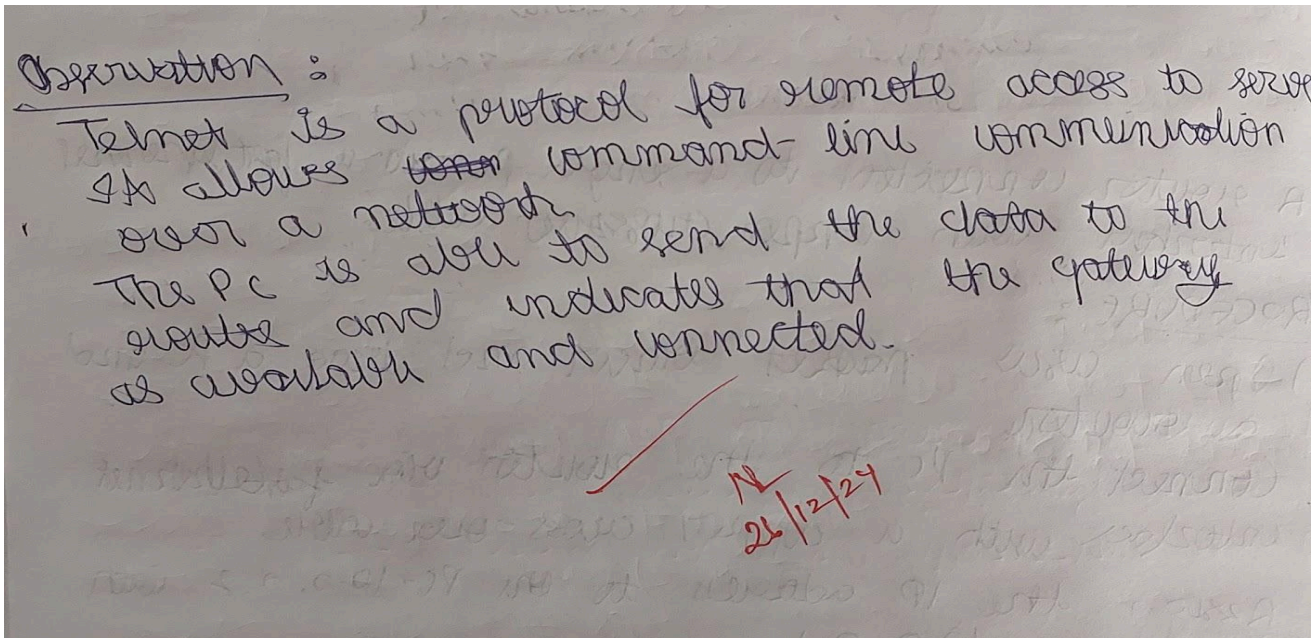
Screen shots/ output:




Router0
Physical
Config
CLI
IOS Command Line Interface
Router(config-if)#exit
Router(config)#hostname R1
R1(config)#enable secret rp
R1(config)#int g0/0/0
% Invalid input detected at '^' marker.
R1(config)#
R1(config)#line vty 0 5
R1(config-line)#login
% Login disabled on line 132, until 'password' is set
% Login disabled on line 133, until 'password' is set
% Login disabled on line 134, until 'password' is set
% Login disabled on line 135, until 'password' is set
% Login disabled on line 136, until 'password' is set
% Login disabled on line 137, until 'password' is set
R1(config-line)#password tp
R1(config-line)#exit
R1(config)#
R1(config)#
R1(config)#exit
R1#
%SYS-5-CONFIG_I: Configured from console by console
R1#wr
Building configuration...
[OK]
R1#
R1#
Copy
Paste


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.2
Pinging 10.0.0.2 with 32 bytes of data:
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=2ms 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 = 2ms, Average = 0ms
PC>telnet 10.0.0.2
Trying 10.0.0.2 ...Open
User Access Verification
Password:
R1>enable
Password:
R1#

Observation:

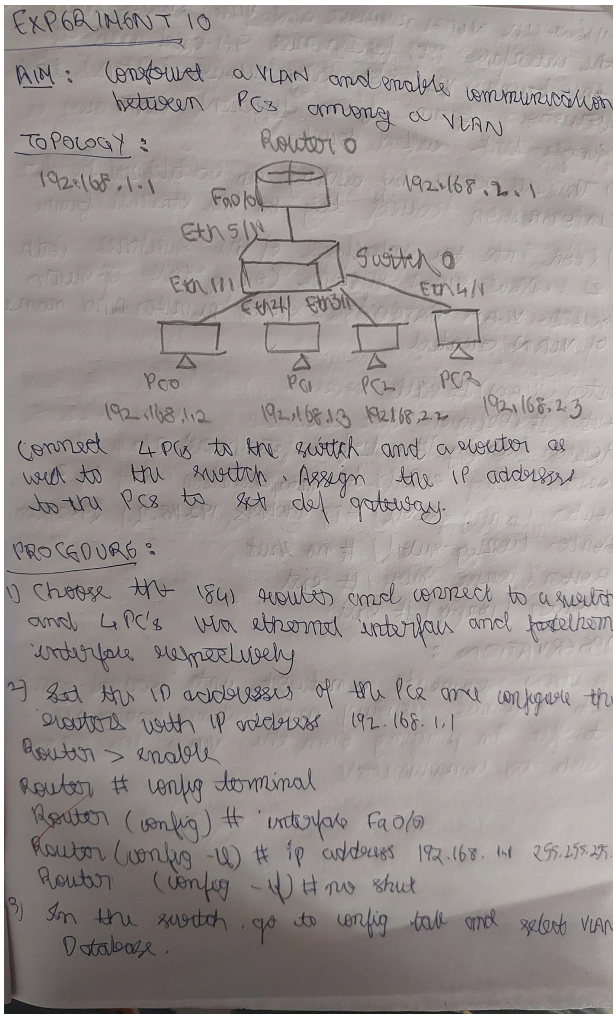

Observation :
Telnet is a protocol for remote access to server
It allows command-line communication
over a network.
The PC is able to send the data to the
router and indicates that the gateway
is available and connected.
21/12/24

Program 11

Aim of the program:

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

Procedure along with the topology:


EXPERIMENT 10

AIM: Construct a VLAN and enable communication between PCs among a VLAN

TOPOLOGY:

Connect 4 PCs to the switch and a router as well to the switch. Assign the IP addresses to the PCs to set def gateway.

PROCEDURE:

1) Choose the 1841 router and connect to a switch and 4 PC's via ethernet interface and fastethernet interface respectively

2) Set the IP addresses of the PCs and configure the routers with IP address 192.168.1.1

Router > enable

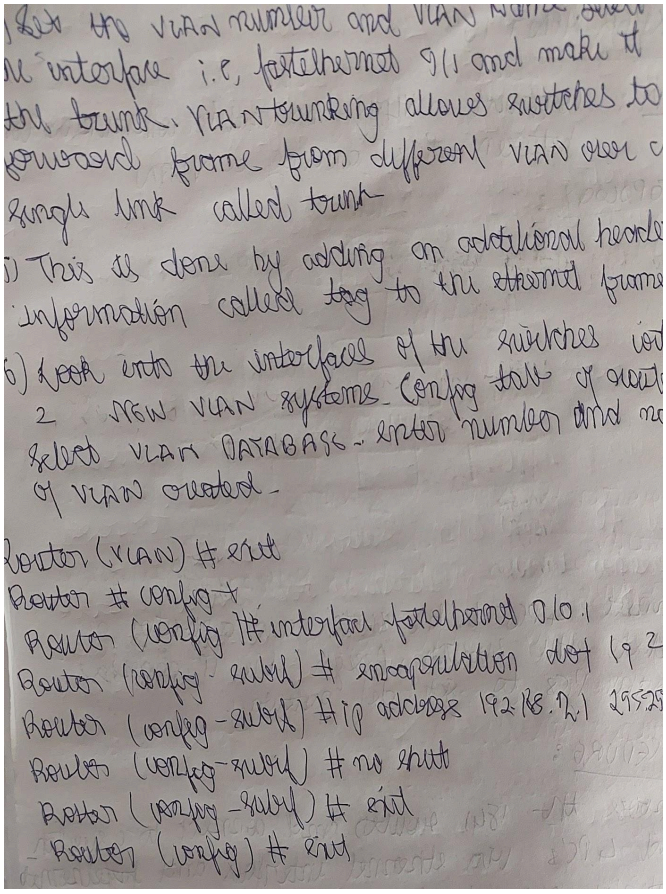
Router # config terminal

Router (config) # interface Fa0/0

Router (config-if) # ip address 192.168.1.1 255.255.255.

Router (config-if) # no shut

3) In the switch, go to config tab and select VLAN Database.


Set the VLAN number and VLAN name. Select the interface i.e, fastethernet 0/1 and make it as trunk. VLAN trunking allows switches to forward frame from different VLAN over a single link called trunk

4) This is done by adding an additional header information called tag to the ethernet frame

5) Look into the interfaces of the switches for 2 new VLAN systems. Config tab of switch. Select VLAN DATABASE. Enter number and name of VLAN created.

Router (vlan) # exit

Router # config t

Router (config) # interface fastethernet 0/0.1

Router (config-subif) # encapsulation dot1q 2

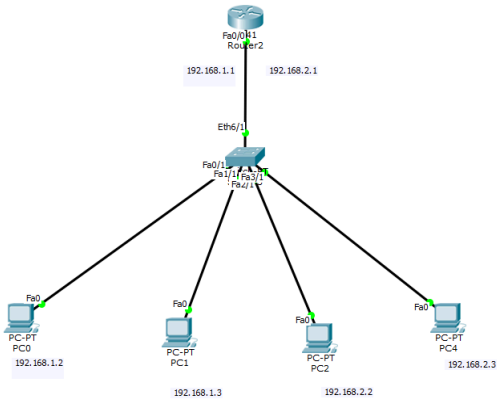
Router (config-subif) # ip address 192.168.1.1 255.25

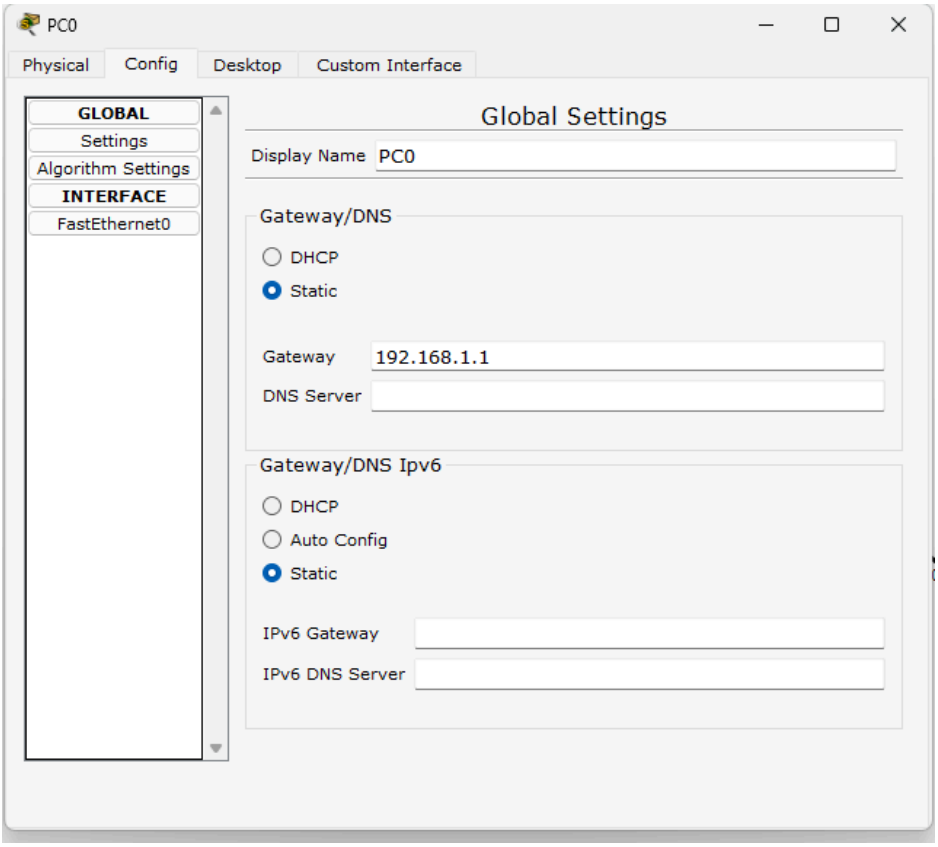
Router (config-subif) # no shut

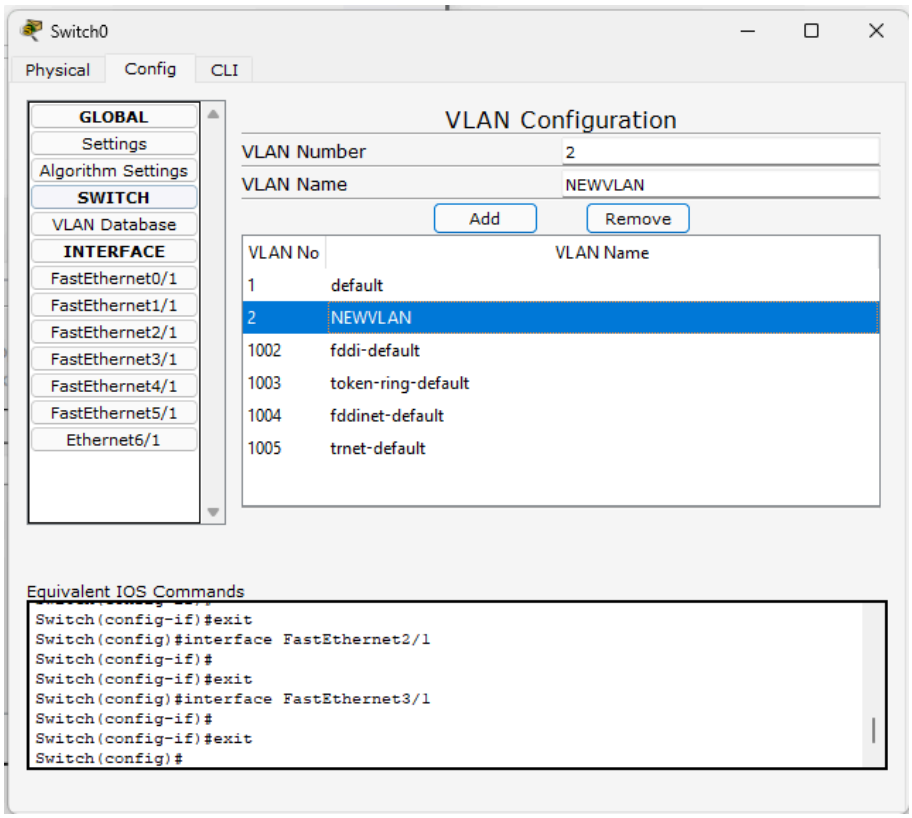
Router (config-subif) # exit

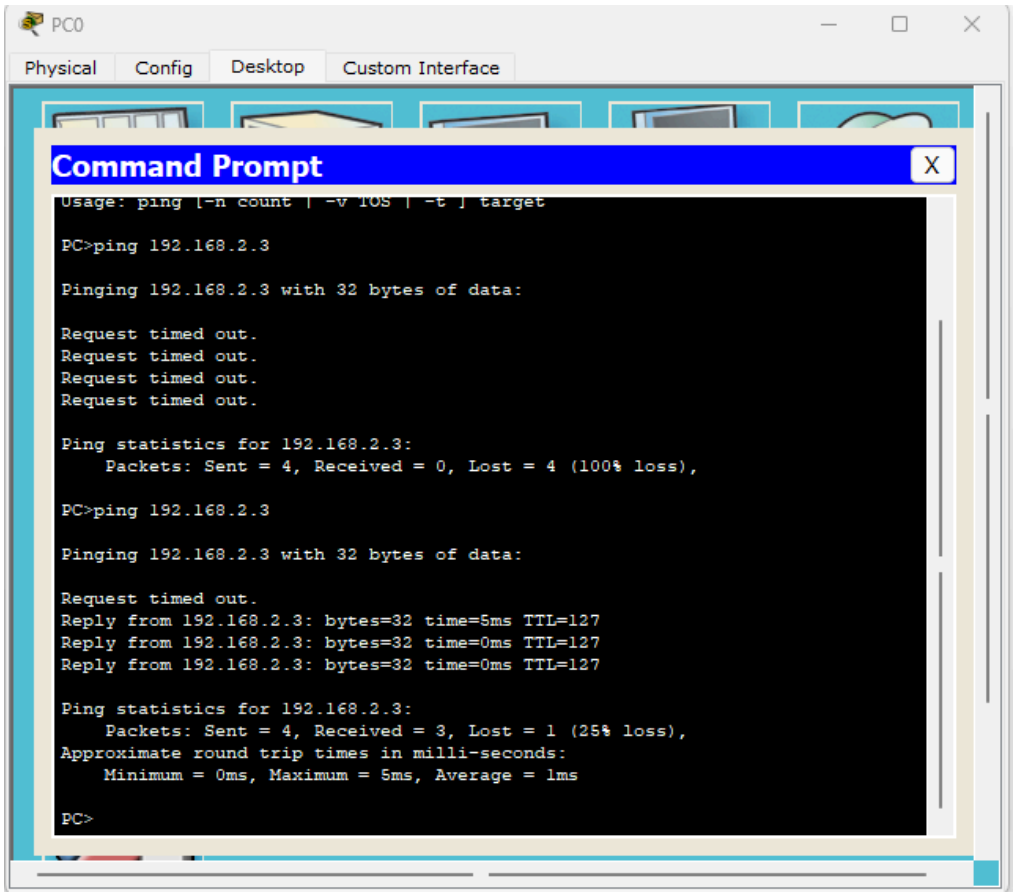
Router (config) # exit

Screen shots/ output:

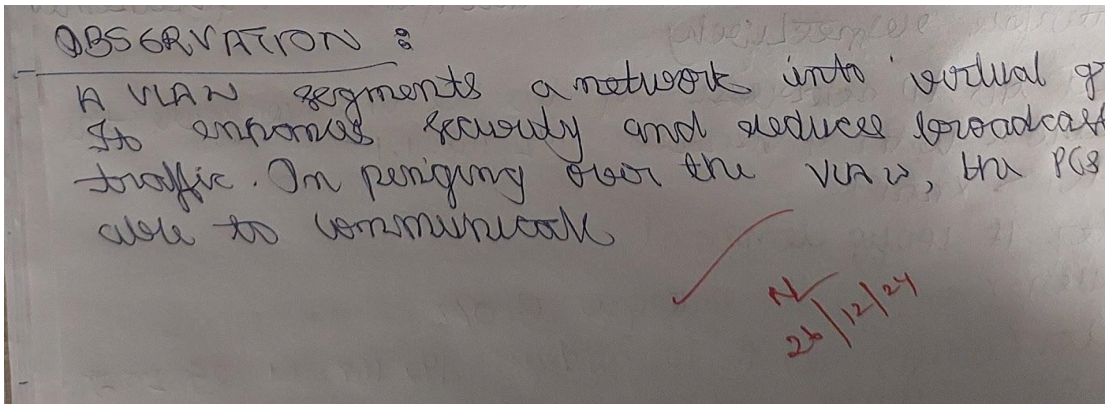





Switch0
Physical
Config
CLI
GLOBAL
Settings
Algorithm Settings
SWITCH
VLAN Database
INTERFACE
FastEthernet0/1
FastEthernet1/1
FastEthernet2/1
FastEthernet3/1
FastEthernet4/1
FastEthernet5/1
Ethernet6/1
VLAN Configuration
VLAN Number
2
VLAN Name
NEWVLAN
Add
Remove
VLAN No
VLAN Name
1 default
2 NEWVLAN
1002 fddi-default
1003 token-ring-default
1004 fddinet-default
1005 trnet-default
Equivalent IOS Commands
Switch(config-if)#exit
Switch(config)#interface FastEthernet2/1
Switch(config-if)#
Switch(config-if)#exit
Switch(config)#interface FastEthernet3/1
Switch(config-if)#
Switch(config-if)#exit
Switch(config)#


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
Usage: ping [-n count | -v TOS | -t] target
PC>ping 192.168.2.3
Pinging 192.168.2.3 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 192.168.2.3:
Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>ping 192.168.2.3
Pinging 192.168.2.3 with 32 bytes of data:
Request timed out.
Reply from 192.168.2.3: bytes=32 time=5ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Reply from 192.168.2.3: bytes=32 time=0ms TTL=127
Ping statistics for 192.168.2.3:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 5ms, Average = 1ms
PC>

Observation:


OBSERVATION :

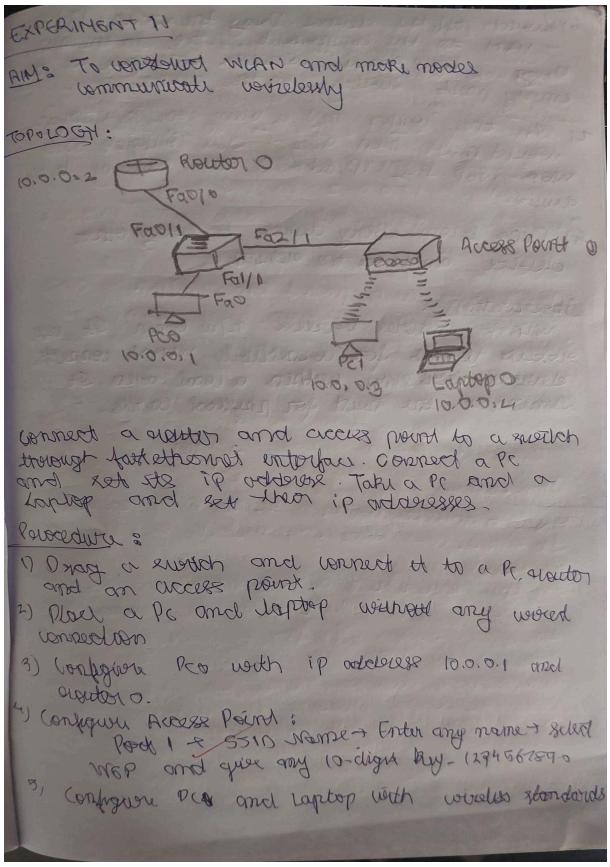
A VLAN segments a network into virtual g
It enhances security and reduces broadcas
traffic. In assigning ports the VLANs, the PCs
able to communicate

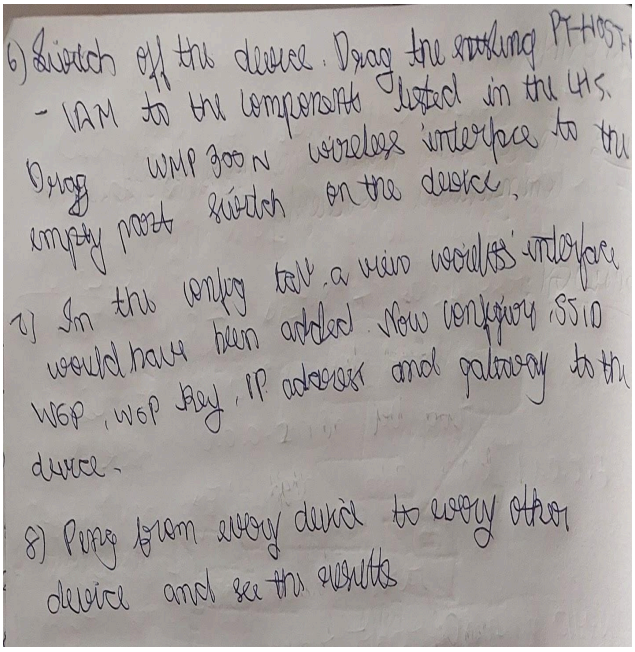
NL
22/12/24

Program 12

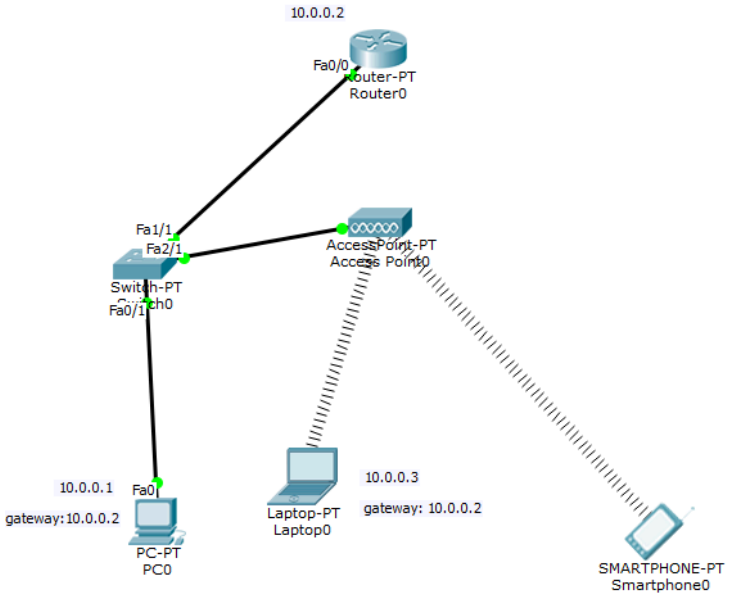
Aim of the program:
To construct a WLAN and make the nodes communicate wirelessly

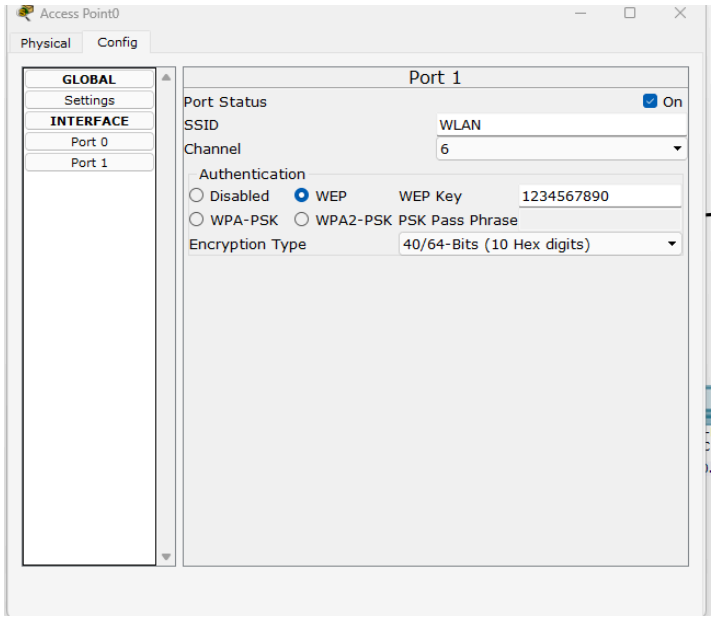
Procedure along with the topology:

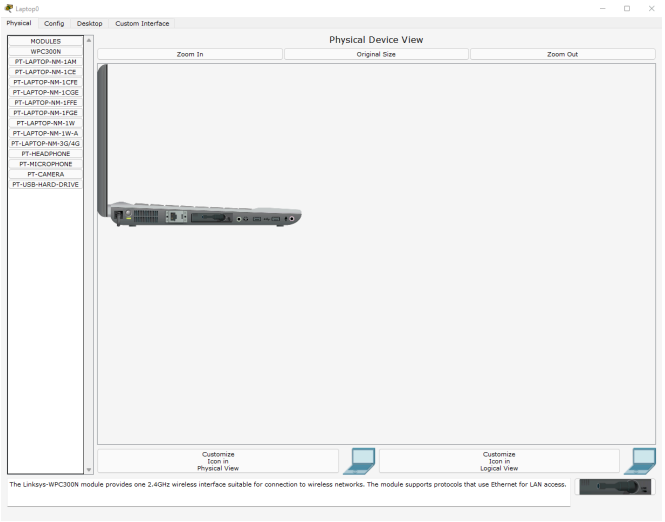


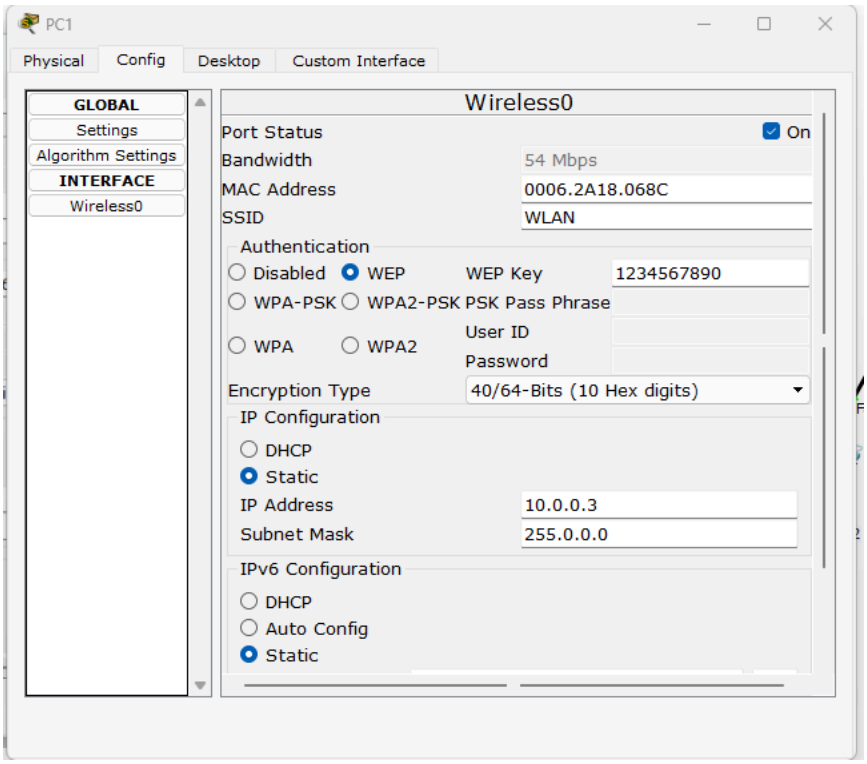


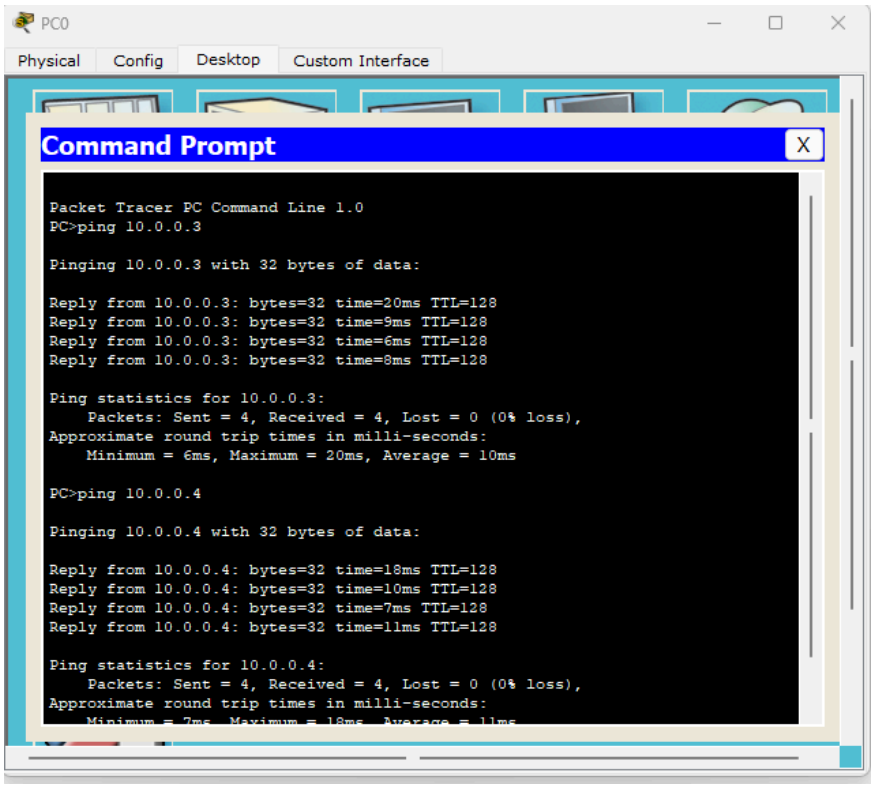
Screen shots/ output :



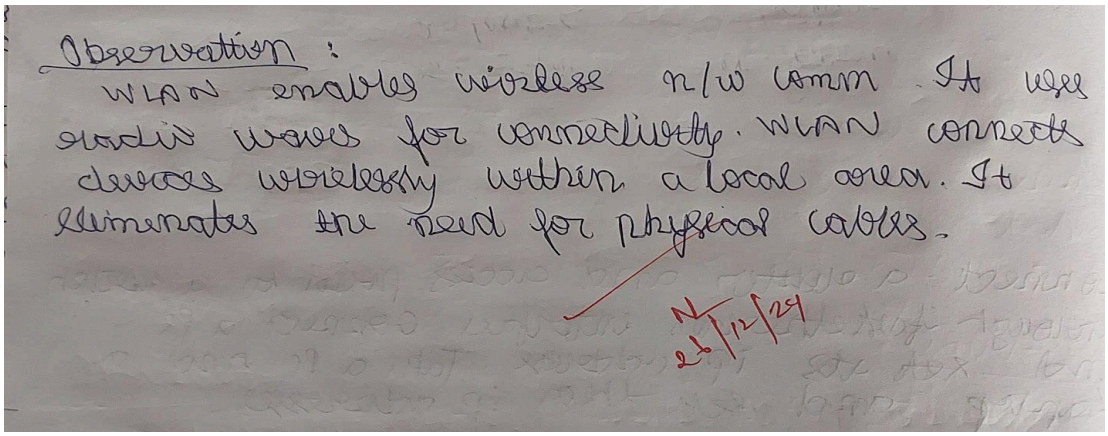

Access Point0
Physical
Config
GLOBAL
Settings
INTERFACE
Port 0
Port 1
Port 1
Port Status
On
SSID
WLAN
Channel
6
Authentication
Disabled
WEP
WEP Key
1234567890
WPA-PSK
WPA2-PSK
PSK Pass Phrase
Encryption Type
40/64-Bits (10 Hex digits)


Physical
Config
Desktop
Custom Interface
MODULES
WPC300N
Physical Device View
Zoom In
Original Size
Zoom Out
Customize Icon in Physical View
Customize Icon in Logical View


PC1
Physical
Config
Desktop
Custom Interface
GLOBAL
Settings
Algorithm Settings
INTERFACE
Wireless0
Wireless0
Port Status
On
Bandwidth
54 Mbps
MAC Address
0006.2A18.068C
SSID
WLAN
Authentication
Disabled
WEP
WEP Key
1234567890
WPA-PSK
WPA2-PSK
PSK Pass Phrase
WPA
WPA2
User ID
Password
Encryption Type
40/64-Bits (10 Hex digits)
IP Configuration
DHCP
Static
IP Address
10.0.0.3
Subnet Mask
255.0.0.0
IPv6 Configuration
DHCP
Auto Config
Static


PC0
Physical
Config
Desktop
Custom Interface
Command Prompt
X
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=20ms TTL=128
Reply from 10.0.0.3: bytes=32 time=9ms TTL=128
Reply from 10.0.0.3: bytes=32 time=6ms TTL=128
Reply from 10.0.0.3: bytes=32 time=8ms 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 = 6ms, Maximum = 20ms, Average = 10ms
PC>ping 10.0.0.4
Pinging 10.0.0.4 with 32 bytes of data:
Reply from 10.0.0.4: bytes=32 time=18ms TTL=128
Reply from 10.0.0.4: bytes=32 time=10ms TTL=128
Reply from 10.0.0.4: bytes=32 time=7ms TTL=128
Reply from 10.0.0.4: bytes=32 time=11ms TTL=128
Ping statistics for 10.0.0.4:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:

Observation:

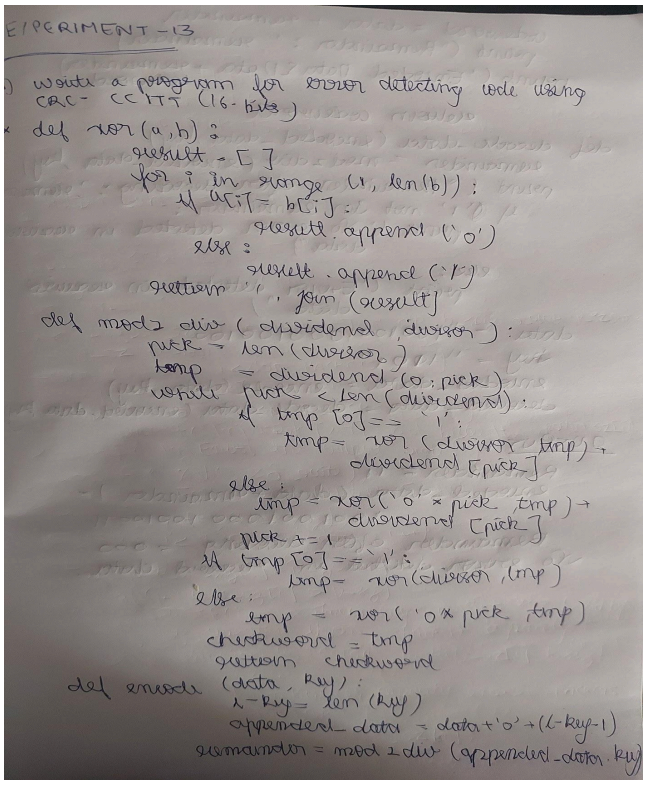

Observation :
WLAN enables wireless n/w comm. It use radio waves for connectivity. WLAN connects devices wirelessly within a local area. It eliminates the need for physical cables.

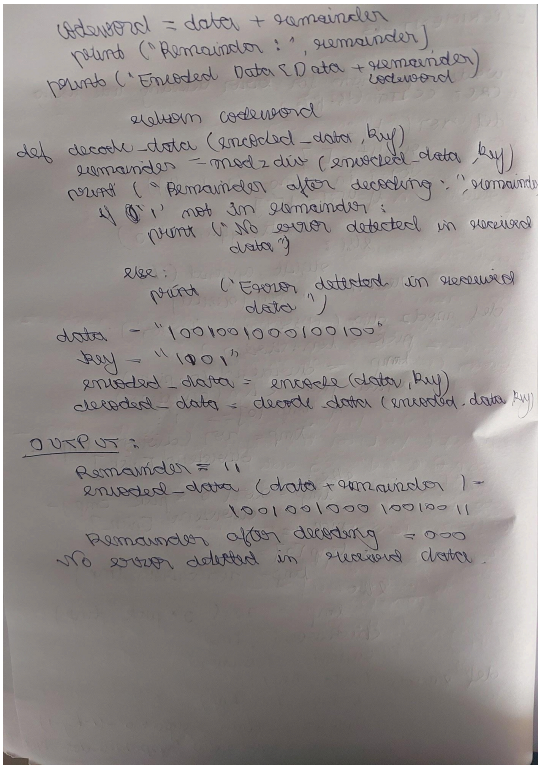
CYCLE-2

Program 1

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

Observation Book:





Code:

```
#include<stdio.h>
#include<string.h>
#define N strlen(gen_poly)

char data[28], check_value[28], gen_poly[10];
int data_length,i,j;

void XOR(){
   for(j = 1;j < N; j++)
   check_value[j] = (( check_value[j] == gen_poly[j])?'0':'1');
}

void receiver(){

   printf("Enter the received data: ");
   scanf("%s", data);
```

```
    printf("\n----------------------------\n");
    printf("Data received: %s", data);

    crc();

    for(i=0;(i<N-1) && (check_value[i]!='1');i++);
        if(i<N-1)
            printf("\nError detected\n\n");
        else
            printf("\nNo error detected\n\n");
}


void crc(){
    for(i=0;i<N;i++)
        check_value[i]=data[i];
    do{
        if(check_value[0]=='1')
            XOR();
        for(j=0;j<N-1;j++)
            check_value[j]=check_value[j+1];
        check_value[j]=data[i++];
    }while(i<=data_length+N-1);
}

int main()
{
    printf("\nEnter data to be transmitted: ");
    scanf("%s",data);
    printf("\n Enter the Generating polynomial: ");
    scanf("%s",gen_poly);
    data_length=strlen(data);
    for(i=data_length;i<data_length+N-1;i++)
        data[i]='0';
    printf("\n----------------------------------------");
    printf("\n Data padded with n-1 zeros : %s",data);
    printf("\n----------------------------------------");
    crc();
    printf("\nCRC or Check value is : %s",check_value);
    for(i=data_length;i<data_length+N-1;i++)
        data[i]=check_value[i-data_length];
    printf("\n----------------------------------------");
    printf("\n Final data to be sent : %s",data);
    printf("\n----------------------------------------\n");
    receiver();
        return 0;
}
```

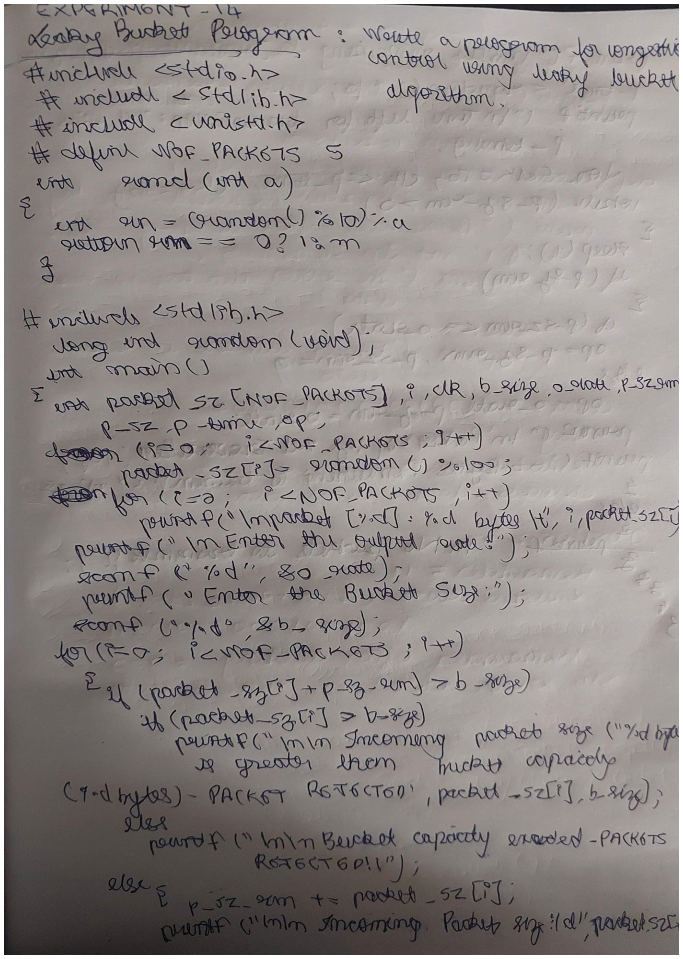
OUTPUT:

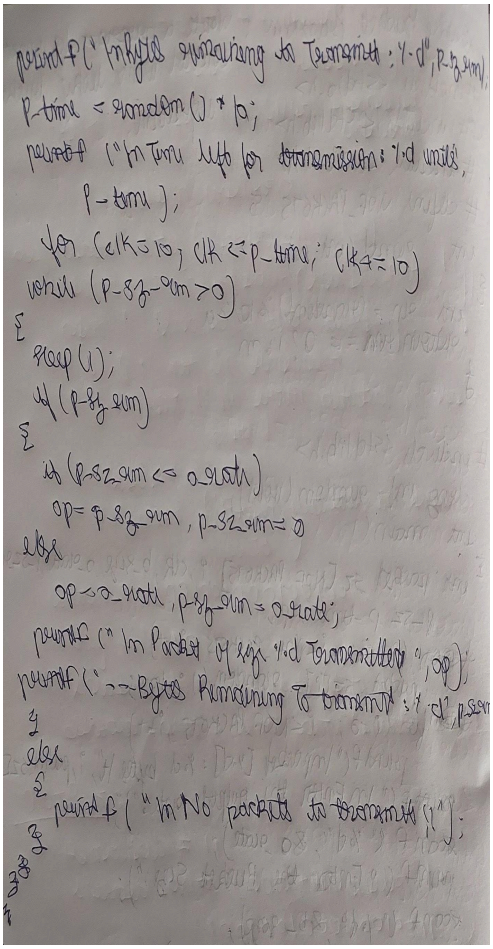
```
Enter data to be transmitted: 1001100
Enter the Generating polynomial: 100001011

----------------------------------------
Data padded with n-1 zeros: 100110000000000
CRC or Check value is: 0100010
```

Program 2

Aim of the program:
Write a program for congestion control using Leaky bucket algorithm.

Observation Book:





Code:

```
# initial packets in the bucket
storage = 0

# total no. of times bucket content is checked
no_of_queries = 4

# total no. of packets that can
# be accommodated in the bucket
bucket_size = 10

# no. of packets that enters the bucket at a time
input_pkt_size = 4

# no. of packets that exits the bucket at a time
output_pkt_size = 1
for i in range(0, no_of_queries):  # space left

    size_left = bucket_size - storage
    if input_pkt_size <= size_left:
      # update storage
        storage += input_pkt_size
    else:
        print("Packet loss = ", input_pkt_size)

    print(f"Buffer size= {storage} out of bucket size = {bucket_size}")

    # as packets are sent out into the network, the size of the storage decreases
    storage -= output_pkt_size
```

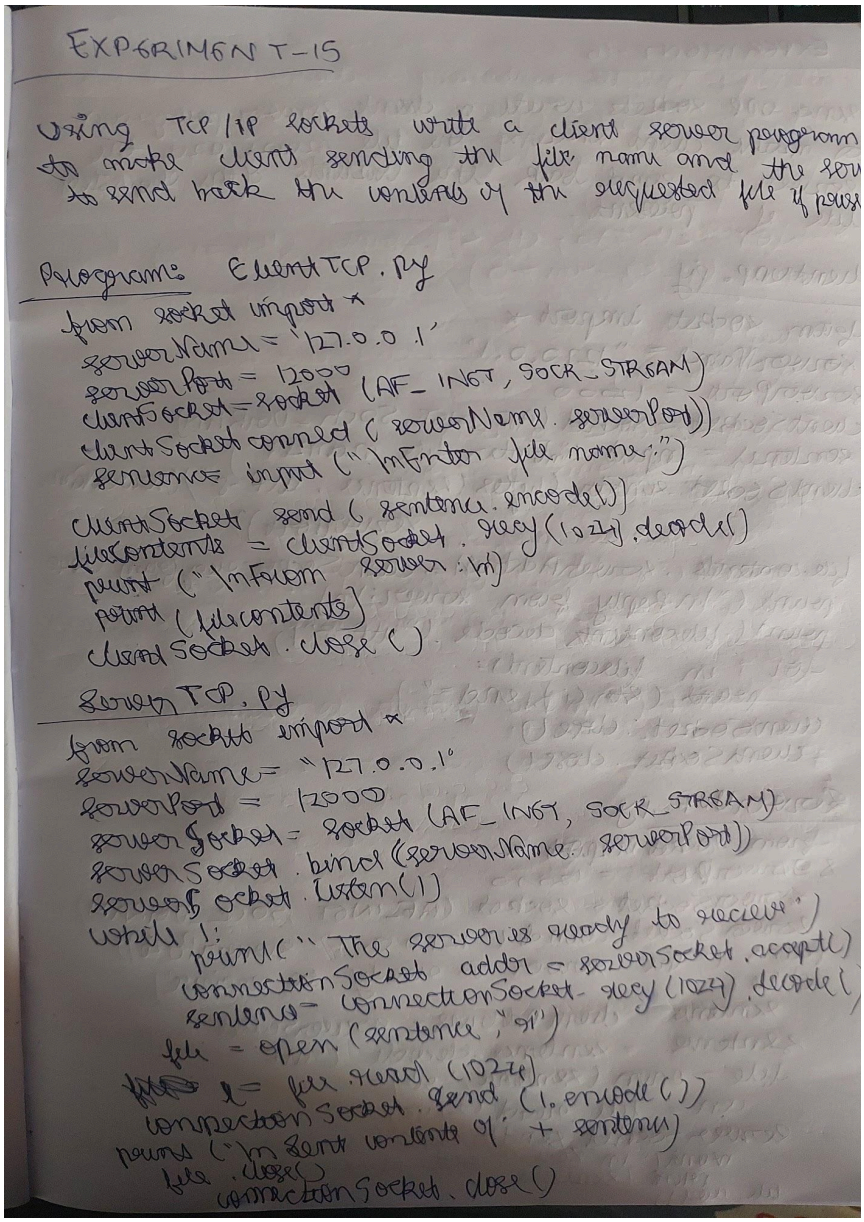
OUTPUT:

```
Enter initial packets in the bucket: 0
Enter total no. of times bucket content is checked: 4
Enter total no. of packets that can be accommodated in the bucket: 10
Enter no. of packets that enters the bucket at a time: 4
Enter no. of packets that exits the bucket at a time: 1
Buffer size = 4 out of bucket size = 10
Buffer size = 7 out of bucket size = 10
Buffer size = 10 out of bucket size = 10
Packet loss = 4
Buffer size = 9 out of bucket size = 10
```

Program 3

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

Observation Book:



EXPERIMENT-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 pres

Program: 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()
```

Code:

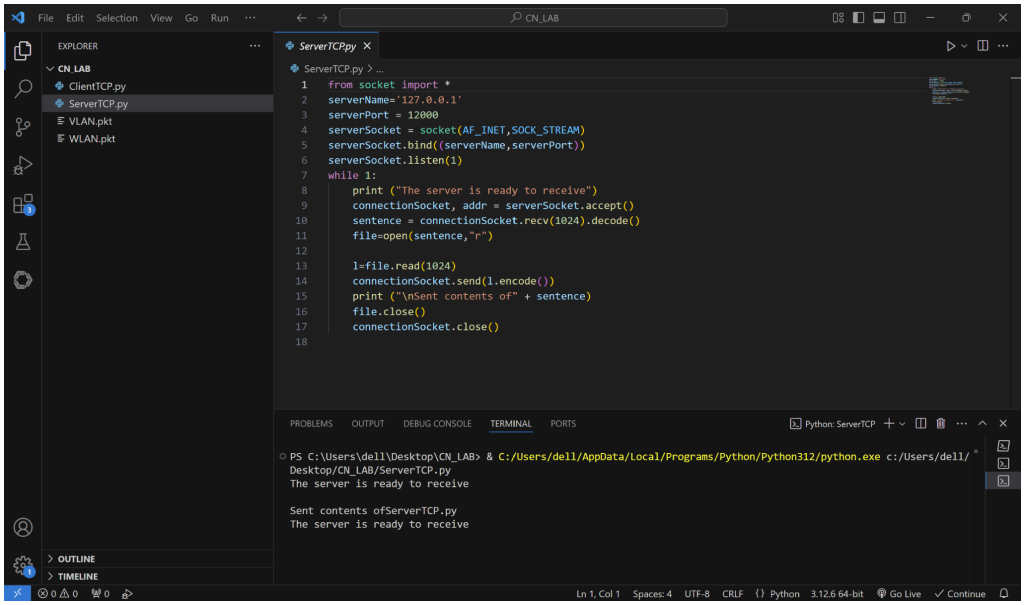
ClientTCP.py

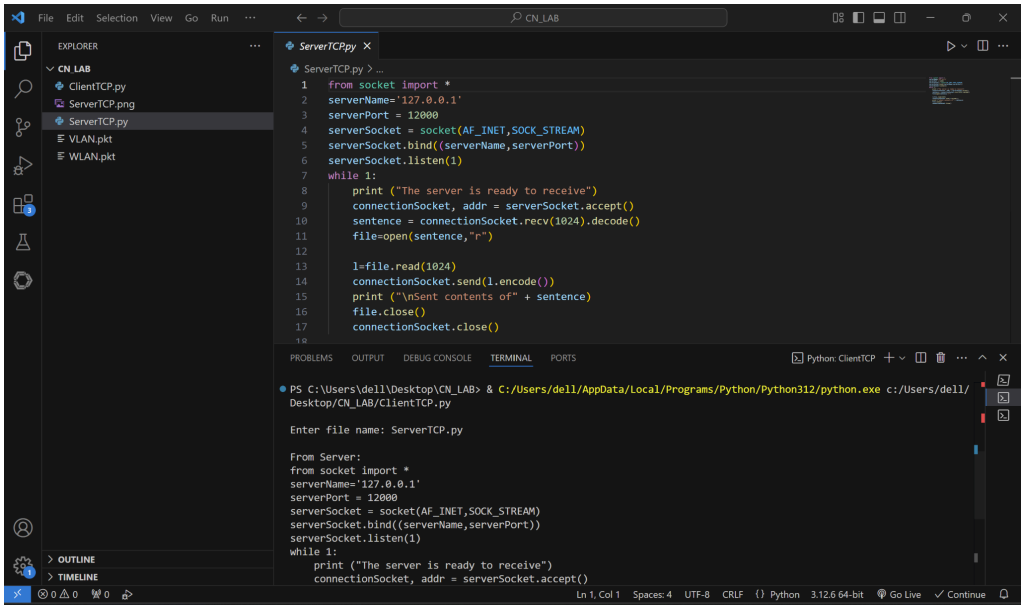
```
from socket import *

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:

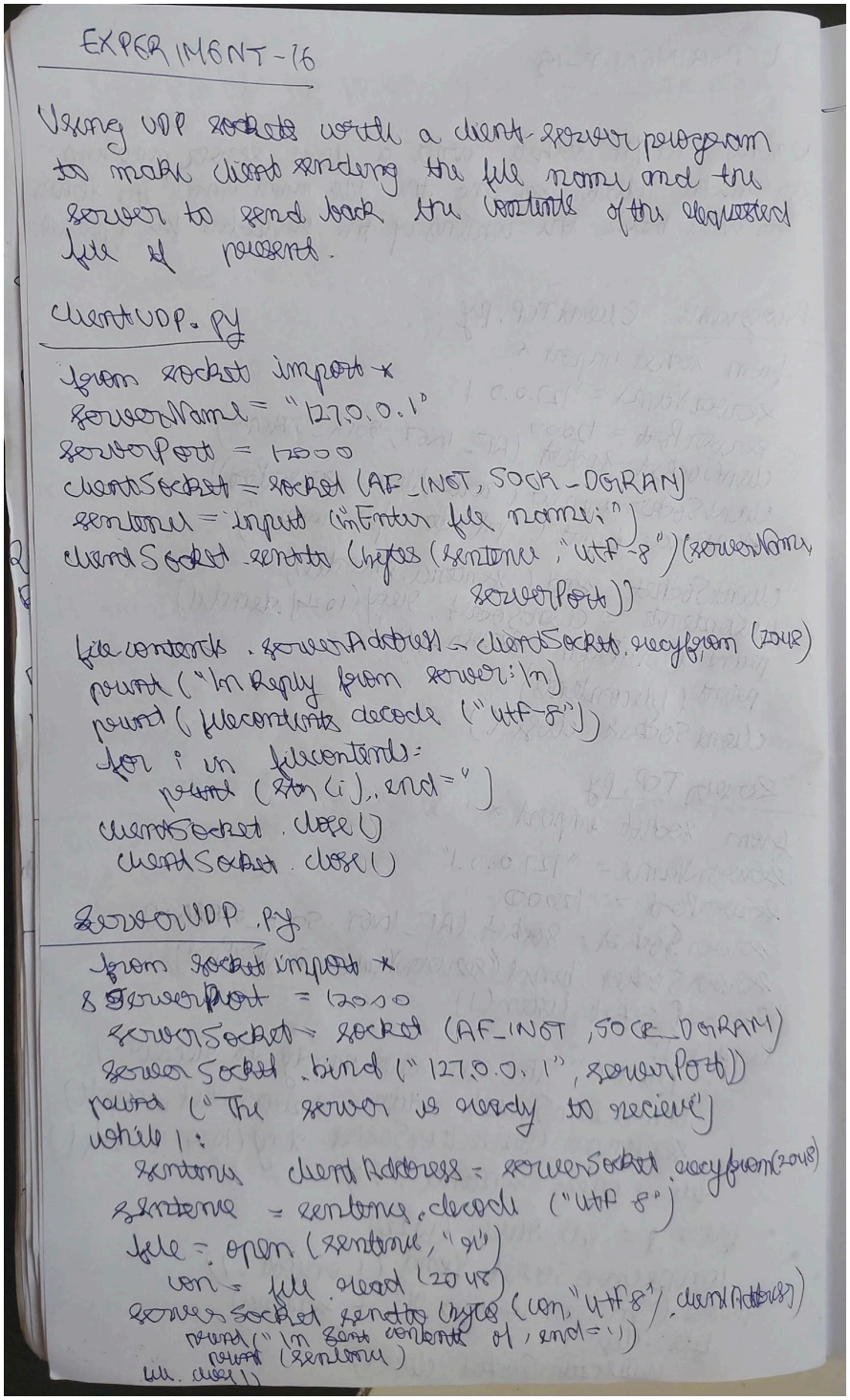

File Edit Selection View Go Run
CN_LAB
EXPLORER
CN_LAB
ClientTCP.py
ServerTCP.py
VLAN.pkt
WLAN.pkt
ServerTCP.py
from socket import *
serverName='127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
print ("The 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()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python: ServerTCP
PS C:\Users\dell\Desktop\CN_LAB> & C:/Users/dell/AppData/Local/Programs/Python/Python312/python.exe c:/Users/dell/Desktop/CN_LAB/ServerTCP.py
The server is ready to receive
Sent contents ofServerTCP.py
The server is ready to receive
OUTLINE
TIMELINE
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.6 64-bit Go Live Continue


File Edit Selection View Go Run
CN_LAB
EXPLORER
CN_LAB
ClientTCP.py
ServerTCP.png
ServerTCP.py
VLAN.pkt
WLAN.pkt
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()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python: ClientTCP
PS C:\Users\dell\Desktop\CN_LAB> & C:/Users/dell/AppData/Local/Programs/Python/Python312/python.exe c:/Users/dell/Desktop/CN_LAB/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 receive")
connectionSocket, addr = serverSocket.accept()
OUTLINE
TIMELINE
Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python 3.12.6 64-bit Go Live Continue

Program 4

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

Observation Book:



EXPERIMENT-16

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.

clientUDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
sentence = input("Enter file name:")
clientSocket.sendto(bytes(sentence,"utf-8"),(serverName,
                    serverPort))

fileContents, serverAddress = clientSocket.recvfrom(2048)
print("\n Reply from server:\n")
print(fileContents.decode("utf-8"))
for i in fileContents:
    print(str(i), end="")
clientSocket.close()
clientSocket.close()
```

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("\n Sent contents of ", end="")
    print(sentence)
    file.close()
```

Code:

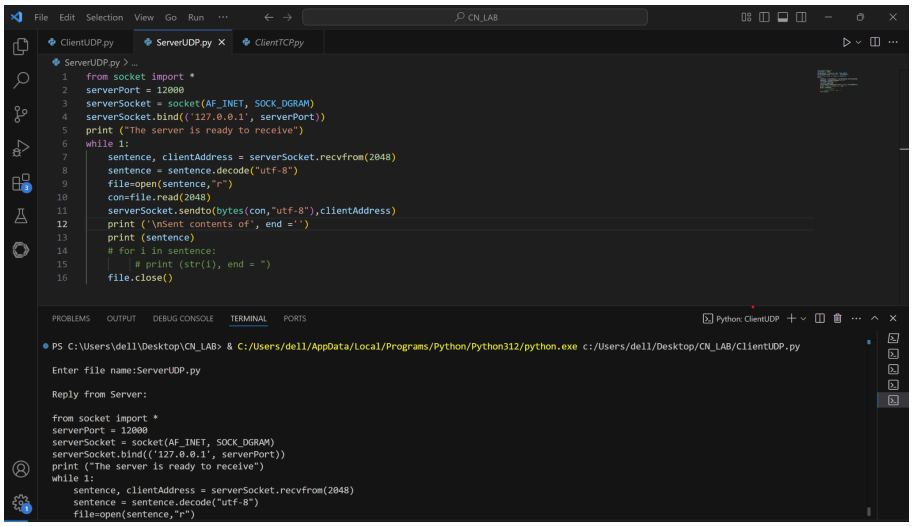
ClientUDP.py

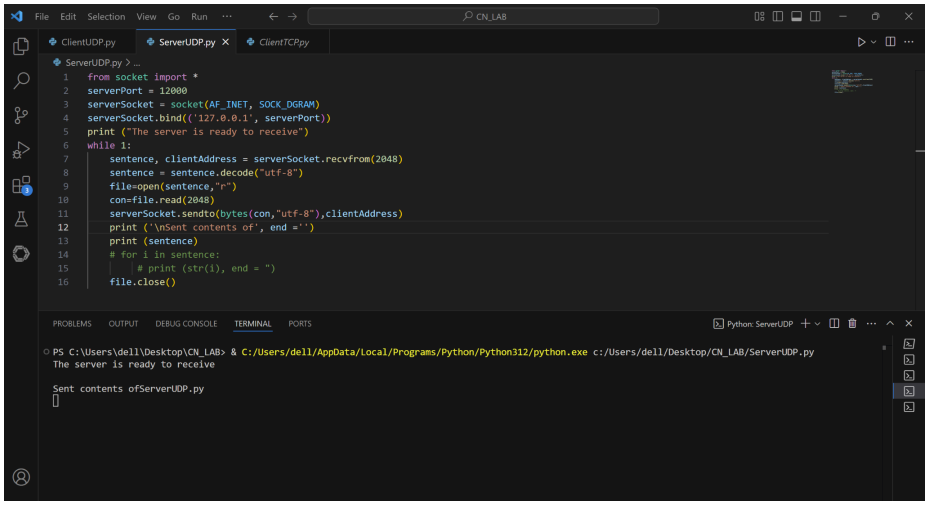
```
from socket import *

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"))
clientSocket.close()
```

ServerUDP.py

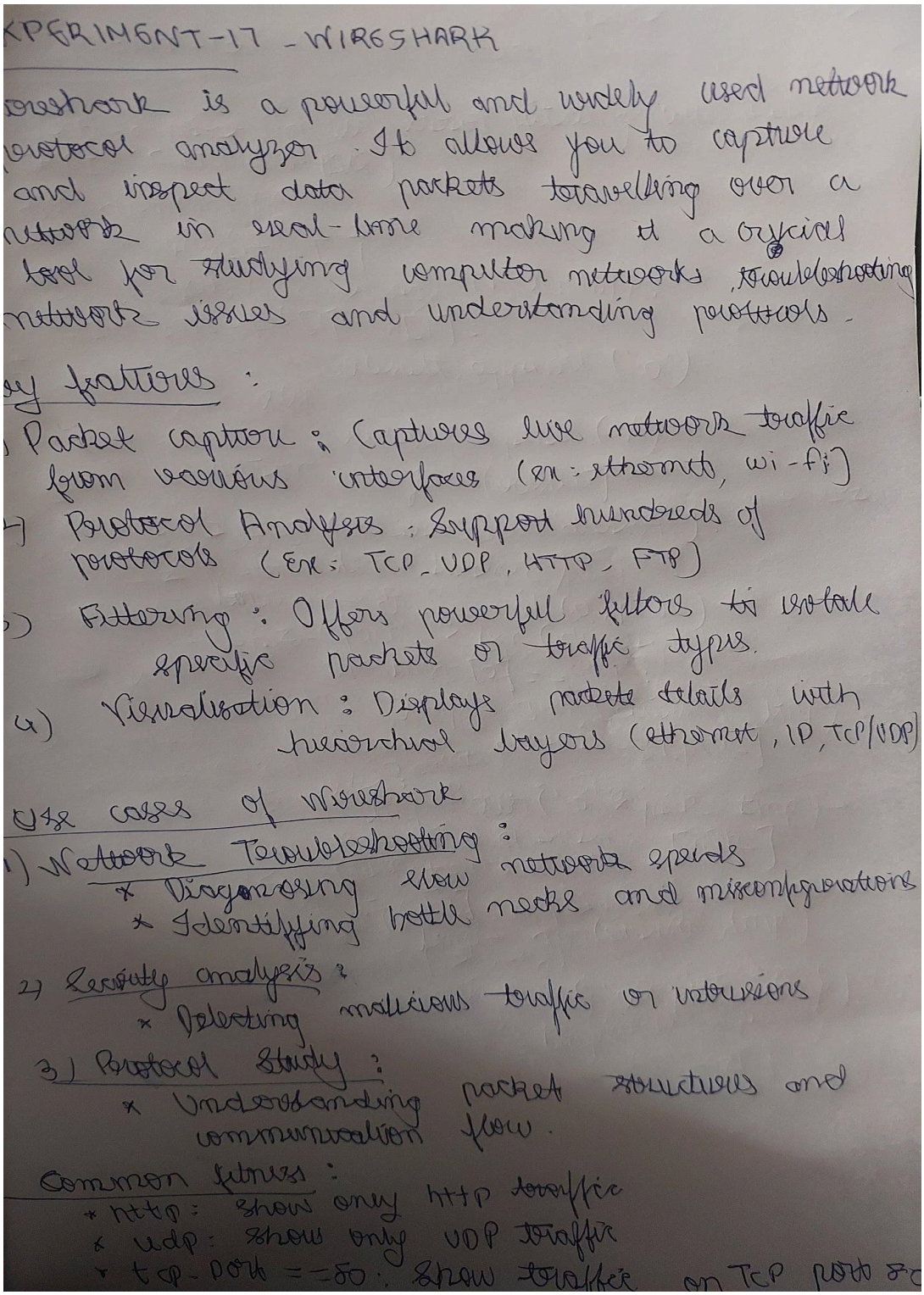
```
from socket import *

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)

    file.close()
```

OUTPUT:


File Edit Selection View Go Run
CN_LAB
ClientUDP.py
ServerUDP.py
ClientTCPpy
ServerUDP.py > ...
1 from socket import *
2 serverPort = 12000
3 serverSocket = socket(AF_INET, SOCK_DGRAM)
4 serverSocket.bind(('127.0.0.1', serverPort))
5 print ("The server is ready to receive")
6 while 1:
7 sentence, clientAddress = serverSocket.recvfrom(2048)
8 sentence = sentence.decode("utf-8")
9 file=open(sentence,"r")
10 con=file.read(2048)
11 serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
12 print ('\nSent contents of', end ='')
13 print (sentence)
14 # for i in sentence:
15 # print (str(i), end = '')
16 file.close()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python: ClientUDP
PS C:\Users\dell\Desktop\CN_LAB> & C:/Users/dell/AppData/Local/Programs/Python/Python312/python.exe c:/Users/dell/Desktop/CN_LAB/ClientUDP.py
Enter file name:ServerUDP.py
Reply from Server:
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")


File Edit Selection View Go Run
CN_LAB
ClientUDP.py
ServerUDP.py
ClientTCPpy
ServerUDP.py > ...
1 from socket import *
2 serverPort = 12000
3 serverSocket = socket(AF_INET, SOCK_DGRAM)
4 serverSocket.bind(('127.0.0.1', serverPort))
5 print ("The server is ready to receive")
6 while 1:
7 sentence, clientAddress = serverSocket.recvfrom(2048)
8 sentence = sentence.decode("utf-8")
9 file=open(sentence,"r")
10 con=file.read(2048)
11 serverSocket.sendto(bytes(con,"utf-8"),clientAddress)
12 print ('\nSent contents of', end ='')
13 print (sentence)
14 # for i in sentence:
15 # print (str(i), end = '')
16 file.close()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python: ServerUDP
PS C:\Users\dell\Desktop\CN_LAB> & C:/Users/dell/AppData/Local/Programs/Python/Python312/python.exe c:/Users/dell/Desktop/CN_LAB/ServerUDP.py
The server is ready to receive
Sent contents ofServerUDP.py

Took Exploration - Wireshark



XPERIMENT-17 - WIRESHARK

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

Key features:

1) Packet capture: Captures live network traffic from various interfaces (ex: ethernet, wi-fi)
2) Protocol Analysis: Support hundreds of protocols (Ex: TCP, UDP, HTTP, FTP)
3) Filtering: Offers powerful filters to isolate specific packets or traffic types.
4) Visualization: Displays packets details with hierarchical layers (ethernet, IP, TCP/UDP)

Use cases of Wireshark

1) Network Troubleshooting:
   * Diagnosing slow network speeds
   * Identifying bottle necks and misconfigurations
2) Security analysis:
   * Detecting malicious traffic or intrusions
3) Protocol Study:
   * Understanding packet structures and communication flow.

Common filters:
* http: show only http traffic
* udp: show only UDP traffic
* tcp.port == 80: show traffic on TCP port 80