VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.

LAB RECORD

Computer Network Lab (23CS5PCCON)

Submitted by

ARCHITA V (1BM23CS050)

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
September 2025 – January 2026

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 **ARCHITA V(1BM23CS045),** 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.

Surabhi S Assistant Professor Department of CSE, BMSCE	**Dr. Kavitha Sooda** Professor & HOD Department of CSE, BMSCE

Index

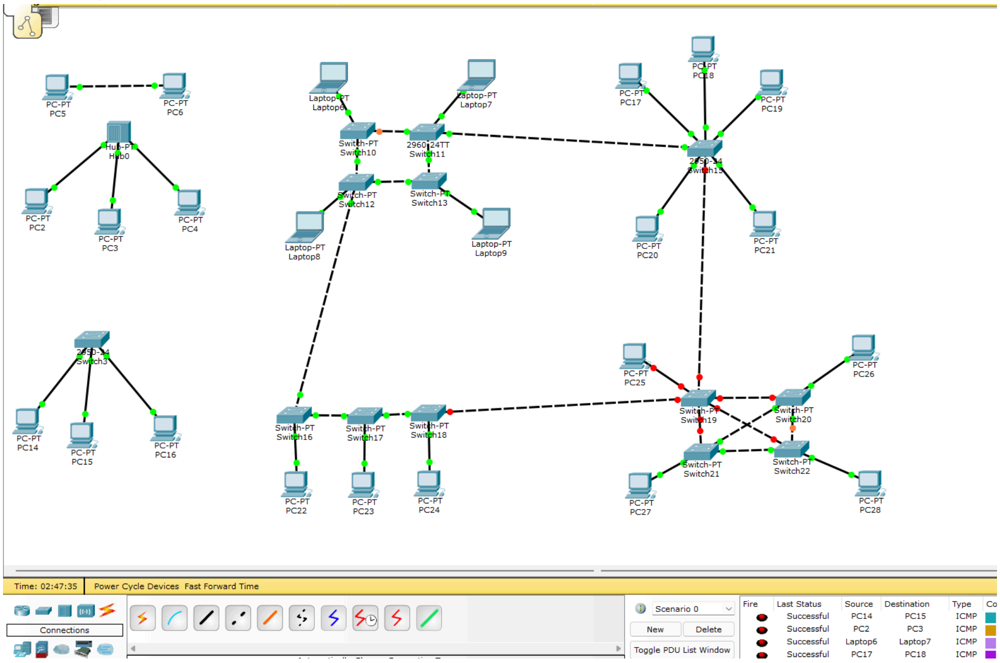
Sl. No.	Experiment Title	Page No.
1	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message	1-3
2	Configure DHCP within a LAN and outside LAN.	4-6
3	Configure Web Server, DNS within a LAN.	7-9
4	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply	10-11
5	Configure default route, static route to the Router	12-16
6	To understand the operation of TELNET by accessing the router in server room from a PC in IT office.	17-19
7	Configure RIP routing Protocol in Routers.	20-22
8	To construct a WLAN and make the nodes communicate wirelessly.	23-24
9	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP).	25-26
10	To construct a VLAN and make the PC's communicate among a VLAN.	27-29
11	Configure OSPF routing protocol.	30-34
12	Write a program for congestion control using Leaky bucket algorithm.	35-36
13	Write a program for error detecting code using CRC-CCITT (16-bits).	37-40
14	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.	41-42
15	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.	43-44

Github Link:
https://github.com/1BM23CS050/CN-Lab

Program – 1:

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

Topology:



Fire	Last Status	Source	Destination	Type	Co
●	Successful	PC14	PC15	ICMP	■
●	Successful	PC2	PC3	ICMP	■
●	Successful	Laptop6	Laptop7	ICMP	■
●	Successful	PC17	PC18	ICMP	■

Procedure:

Point-to-Point link

- **PC5** → 20.10.0.2
- **PC6** → 20.10.0.3

Hub (PC2, PC3, PC4)

- **PC2** → 20.10.0.4
- **PC3** → 20.10.0.5
- **PC4** → 20.10.0.6

Switch (PC14, PC15, PC16)

- **PC14** → 20.10.0.7
- **PC15** → 20.10.0.8
- **PC16** → 20.10.0.9

LAN Topology & PC IP Configurations

1. Star Topology (Switch13 Region)

- **PC17** → 10.0.0.2
- **PC18** → 10.0.0.3
- **PC19** → 10.0.0.4
- **PC20** → 10.0.0.5
- **PC21** → 10.0.0.6

2. Bus Topology (Switch16 → Switch17 → Switch18)

- **PC22** → 10.0.0.7
- **PC23** → 10.0.0.8
- **PC24** → 10.0.0.9

3. Ring Topology (Switch19 → Switch20 → Switch21 → Switch22)

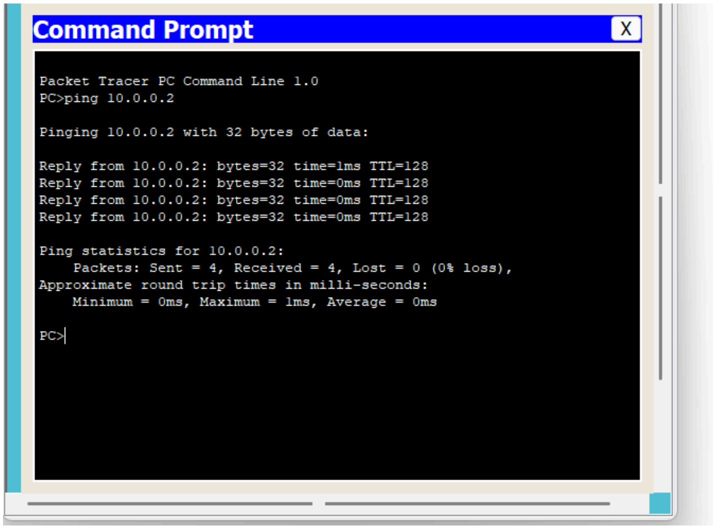
- **PC25** → 10.0.0.10

- **PC26** → 10.0.0.11
- **PC27** → 10.0.0.12
- **PC28** → 10.0.0.13

4. Mesh Topology (Bottom-right interconnected switches)
If mesh includes the same PCs (25–28), then **IPs remain the same**.
If you have more PCs, assign next IPs:

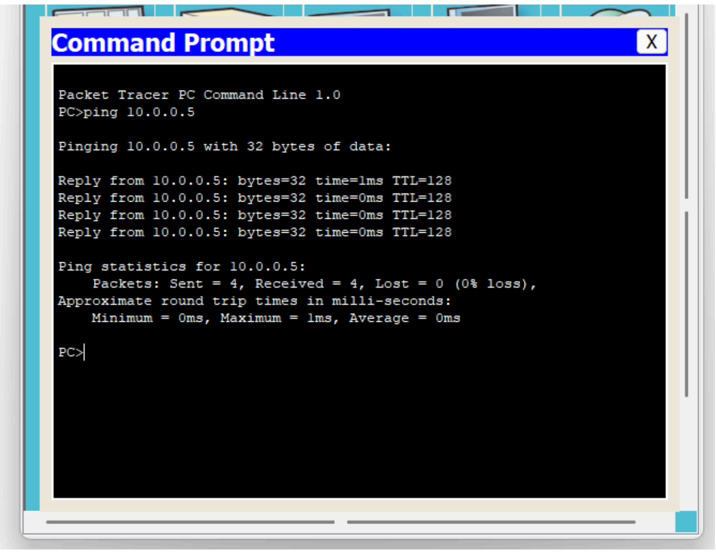
- **Next PC** → 10.0.0.14
- **Next PC** → 10.0.0.15
- Continue as needed…

Output:



```
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=1ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128
Reply from 10.0.0.2: bytes=32 time=0ms TTL=128

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 = 1ms, Average = 0ms

PC>
```

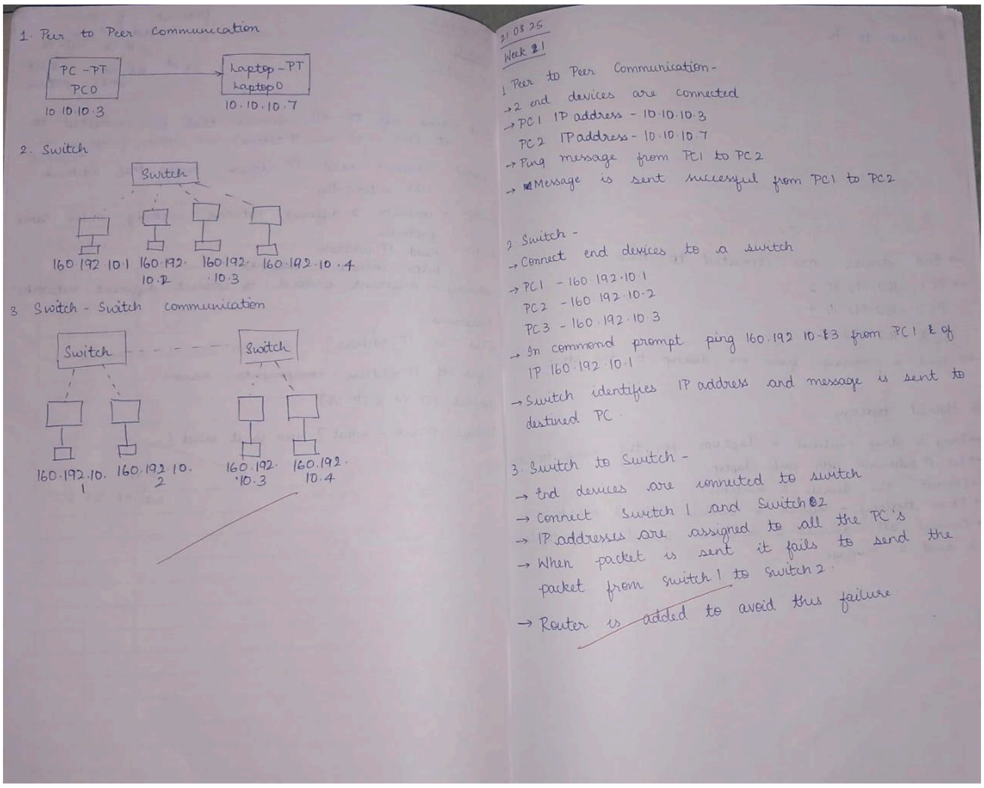
Fig 1.1Command Prompt of PC22



```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.5

Pinging 10.0.0.5 with 32 bytes of data:

Reply from 10.0.0.5: bytes=32 time=1ms TTL=128
Reply from 10.0.0.5: bytes=32 time=0ms TTL=128
Reply from 10.0.0.5: bytes=32 time=0ms TTL=128
Reply from 10.0.0.5: bytes=32 time=0ms TTL=128

Ping statistics for 10.0.0.5:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

PC>
```

Fig 1.2 Command Prompt of PC26

Observation:



1\. Peer to Peer Communication

2\. Switch

3 Switch - Switch communication

21 08 25

Week 1

1 Peer to Peer Communication -

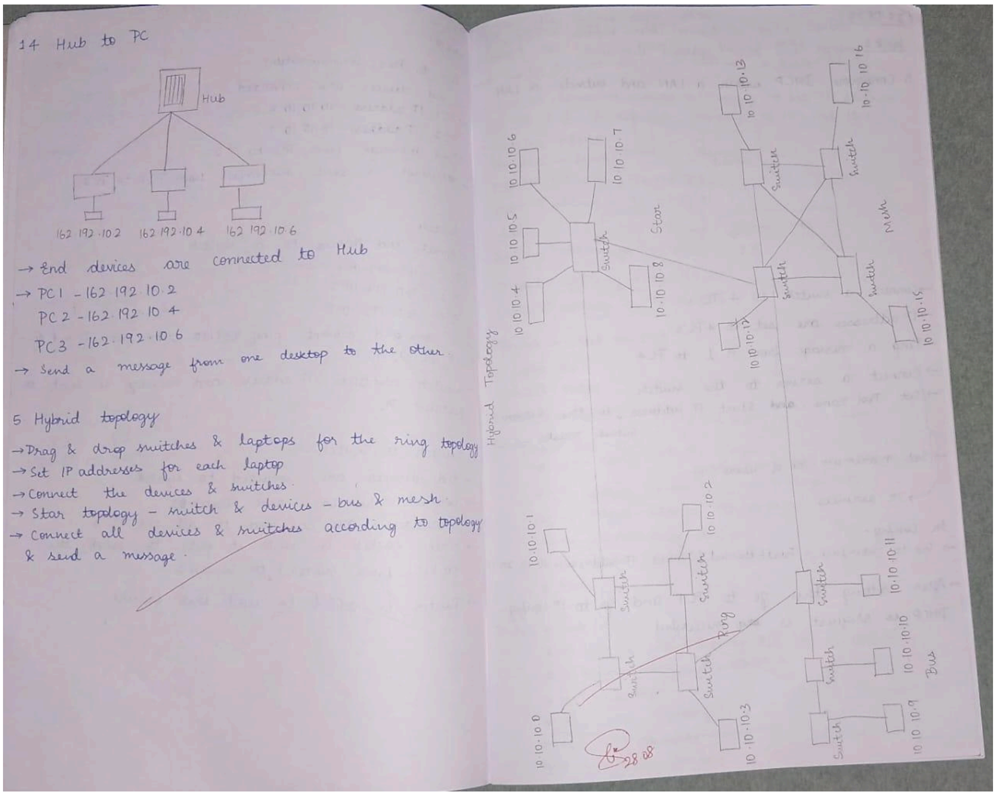
→ 2 end devices are connected
→ PC1 IP address - 10.10.10.3
PC2 IPaddress - 10.10.10.7
→ Ping message from PC1 to PC2
→ Message is sent succesful from PC1 to PC2

2 Switch -

→ Connect end devices to a switch
→ PC1 - 160.192.10.1
PC2 - 160.192.10.2
PC3 - 160.192.10.3
→ In commond prompt ping 160.192.10.3 from PC1 & of IP 160.192.10.1
→ Switch identifies IP address and message is sent to destined PC.

3\. Switch to Switch -

→ End devices are connected to switch
→ Connect Switch 1 and Switch 2
→ IP addresses are assigned to all the PC's
→ When packet is sent it fails to send the packet from switch 1 to switch 2.
→ Router is added to avoid this failure



14 Hub to PC

→ End devices are connected to Hub
→ PC1 - 162.192.10.2
PC2 - 162.192.10.4
PC3 - 162.192.10.6
→ Send a message from one desktop to the other

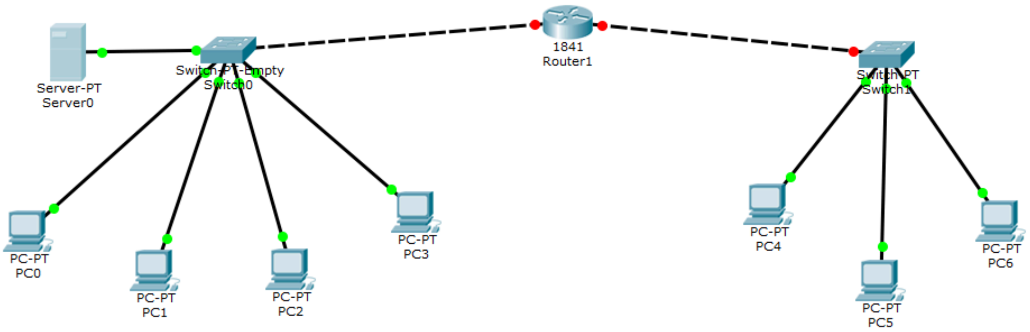
5 Hybrid topology

→ Drag & drop switches & laptops for the ring topology
→ Set IP addresses for each laptop
→ Connect the devices & switches.
→ Star topology - switch & devices - bus & mesh
→ Connect all devices & switches according to topology & send a message.

Program – 2:

Aim: Configure DHCP within a LAN and outside LAN.

Topology:



Procedure:

1. Configure DHCP Server

1. On the Server → **Desktop** > **IP Configuration**, set:
    - **Static IP:** 192.168.10.2
    - **Default Gateway:** 192.168.10.1
2. Open **Services** > **DHCP** and add the following two DHCP pools:

(a) Pool Name: switch1

- Gateway: 192.168.10.1
- Start IP: 192.168.10.3
- Subnet Mask: 255.255.255.0
- Max Users: 20

(b) Pool Name: switch2

- Gateway: 192.168.20.1
- Start IP: 192.168.20.2
- Subnet Mask: 255.255.255.0
- Max Users: 20

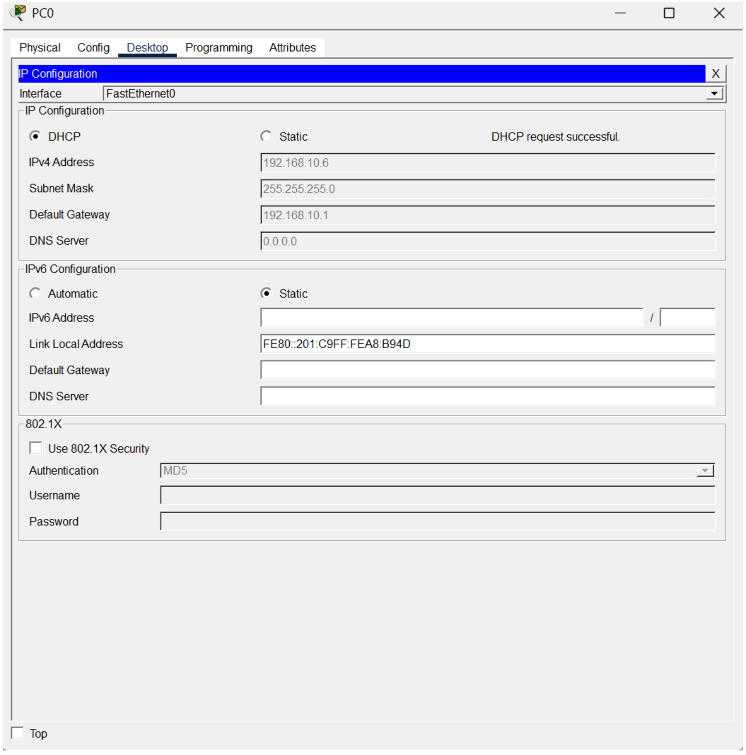
3. Turn **DHCP Service ON**.
4. Go to **Config** > **Interface** > **FastEthernet0** and assign server IP:
    - IP Address: 192.168.10.10
    - Port Status: ✔ ON

2. Configure Router

```
i. Router > enable
ii. Router# configure terminal
(Within LAN)
iii. Router(config)# int fa0/0
iv. Router(config-if)# ip address 192.168.10.1 255.255.255.0
v. Router(config-if)# ip helper-address 192.168.10.2
vi. Router(config-if)# no shutdown
vii. Router(config-if)# exit
(Outside LAN)
viii. Router(config)# int fa0/1
ix. Router(config-if)# ip address 192.168.20.1 255.255.255.0
x. Router(config-if)# ip helper-address 192.168.10.2
xi. Router(config-if)# no shutdown
xii. Router(config-if)# exit
xiii. Router(config)# exit
```

xiv. Router# **write memory**

Output:



IP Configuration

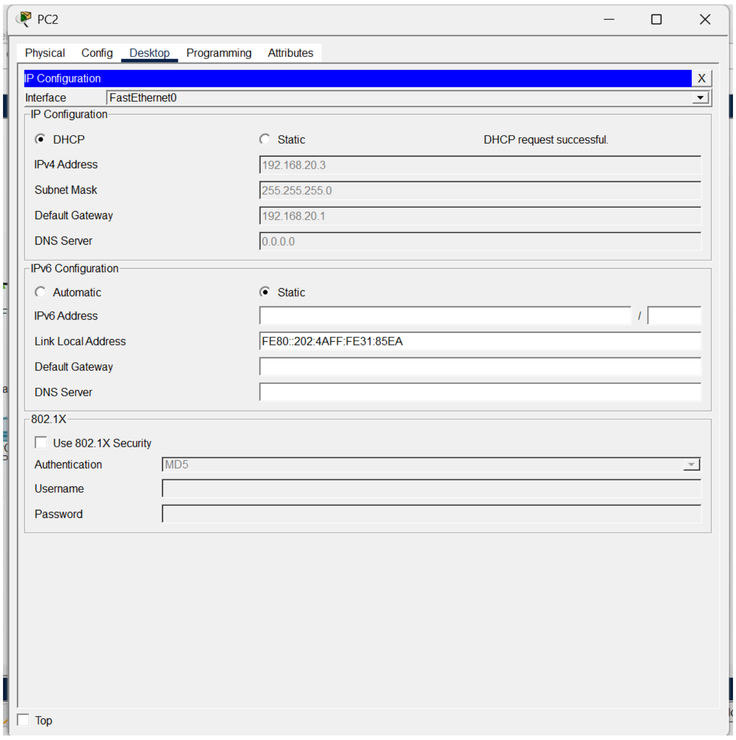
Interface: FastEthernet0

IP Configuration
● DHCP ○ Static DHCP request successful.
IPv4 Address: 192.168.10.6
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.10.1
DNS Server: 0.0.0.0

IPv6 Configuration
○ Automatic ● Static
IPv6 Address: /
Link Local Address: FE80::201:C9FF:FEA8:B94D
Default Gateway:
DNS Server:

802.1X
☐ Use 802.1X Security
Authentication: MD5
Username:
Password:

Fig 2.1 DHCP Inside LAN



IP Configuration

Interface: FastEthernet0

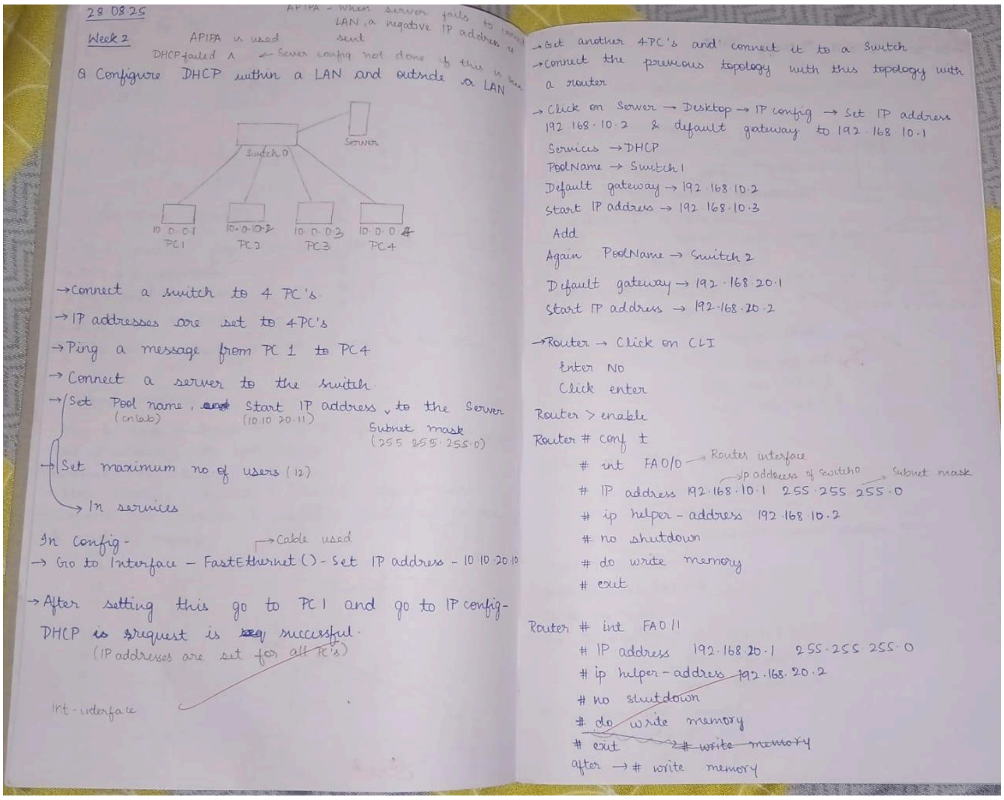
IP Configuration
● DHCP ○ Static DHCP request successful.
IPv4 Address: 192.168.20.3
Subnet Mask: 255.255.255.0
Default Gateway: 192.168.20.1
DNS Server: 0.0.0.0

IPv6 Configuration
○ Automatic ● Static
IPv6 Address: /
Link Local Address: FE80::202:4AFF:FE31:85EA
Default Gateway:
DNS Server:

802.1X
☐ Use 802.1X Security
Authentication: MD5
Username:
Password:

Fig 2.2 DHCP Outside LAN

Observation:



28.08.25

Week 2

APIPA is used
DHCP failed ^ ← Server config not done if this is the

APIPA - When server fails to connect LAN, a negative IP address is sent

Q Configure DHCP within a LAN and outside a LAN

→ Connect a switch to 4 PC's

→ IP addresses are set to 4 PC's

→ Ping a message from PC 1 to PC4

→ Connect a server to the switch

→ Set Pool name (cnlab), ~~and~~ Start IP address (10.10.20.11), Subnet mask (255.255.255.0) to the Server

→ Set maximum no of users (12)

→ In services

In config -

→ Go to Interface - FastEthernet () → Cable used - Set IP address - 10.10.20.10

→ After setting this go to PC 1 and go to IP config - DHCP ~~is~~ request is ~~seq~~ successful.
(IP addresses are set for all PC's)

int - interface

→ Get another 4 PC's and connect it to a Switch

→ Connect the previous topology with this topology with a router

→ Click on Server → Desktop → IP config → Set IP address 192.168.10.2 & default gateway to 192.168.10.1

Services → DHCP

PoolName → Switch 1

Default gateway → 192.168.10.2

Start IP address → 192.168.10.3

Add

Again PoolName → Switch 2

Default gateway → 192.168.20.1

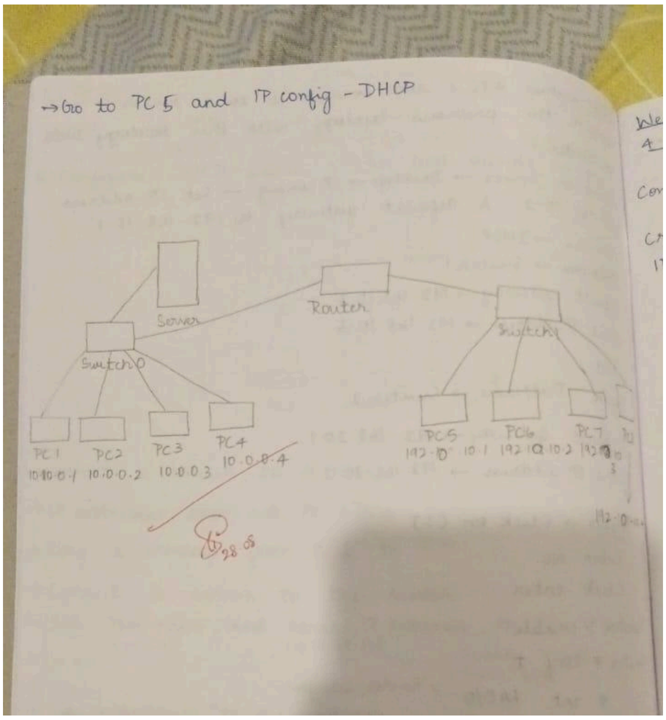
Start IP address → 192.168.20.2

→ Router → Click on CLI

Enter NO

Click enter

```
Router > enable
Router # conf t
        # int FA0/0 → Router interface
        # IP address 192.168.10.1 255.255.255.0   (→ ip address of switch0, → Subnet mask)
        # ip helper-address 192.168.10.2
        # no shutdown
        # do write memory
        # exit

Router # int FA0/1
        # IP address 192.168.20.1 255.255.255.0
        # ip helper-address 192.168.20.2
        # no shutdown
        # do write memory
        # exit          ~~# write memory~~
        after → # write memory
```



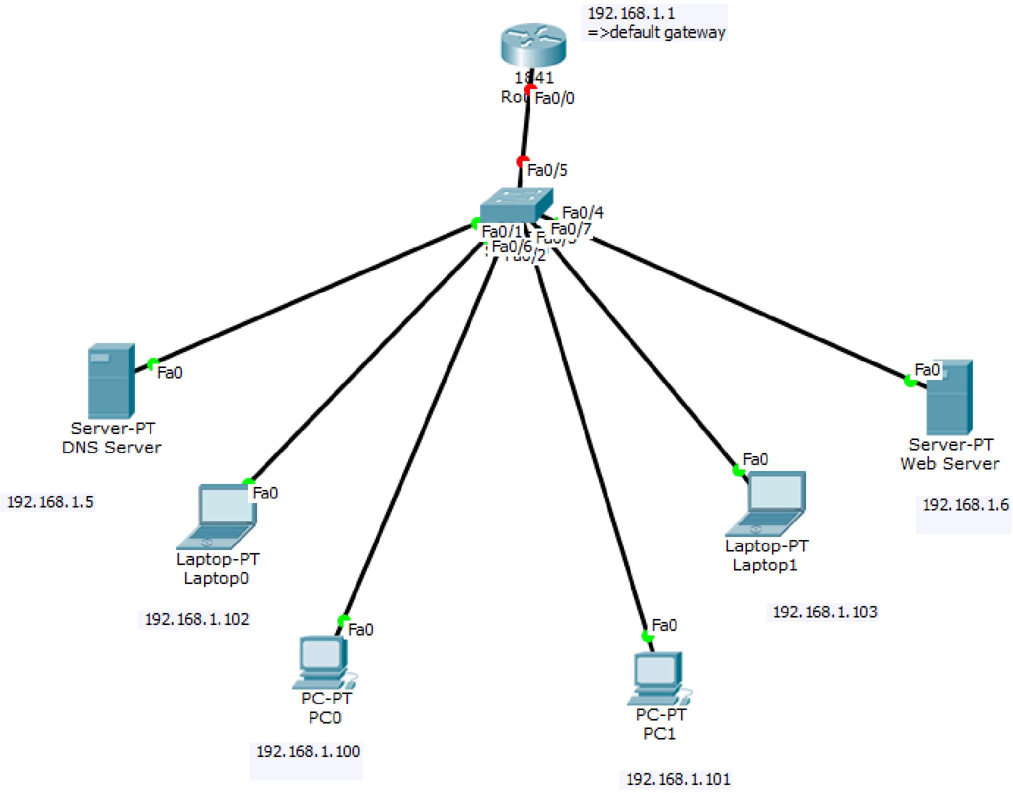
→ Go to PC 5 and IP config - DHCP

28.08

Program – 3:

Aim: Configure Web Server, DNS within a LAN.

Topology:



Procedure:

1. PC Configurations

PC0

- IP Address: **192.168.1.100**
- Subnet Mask: **255.255.255.0**
- Default Gateway: **192.168.1.1**
- DNS: **192.168.1.5**

PC1

- IP Address: **192.168.1.101**
- Subnet Mask: **255.255.255.0**
- Default Gateway: **192.168.1.1**
- DNS: **192.168.1.5**

2. Servers Required

- **DNS Server**
- **Web Server**

3. Configure Web Server

1. Open **Server > Services > HTTP**
2. Turn **HTTP** and **HTTPS ON**
3. Go to **Desktop > Edit HTML**
    - Edit the default HTML page to display a custom message
4. Assign IP to Web Server:

        - **IP:** 192.168.1.6
        - **Subnet Mask:** 255.255.255.0
        - **Default Gateway:** 192.168.1.1
        - **DNS:** 192.168.1.5

4. Configure DNS Server

1. Go to **Services** > **DNS**
2. Add a new DNS entry:
    - **Name:** www
    - **Domain: letslearn.com**
    - **Address:** 192.168.1.6 (IP of web server)
3. Ensure DNS Service is **ON**
4. DNS Server IP configuration:
    - **IP:** 192.168.1.5
    - **Subnet Mask:** 255.255.255.0
    - **Default Gateway:** 192.168.1.1
    - **DNS:** 192.168.1.5

5. Check Connectivity

On PC → **Command Prompt**

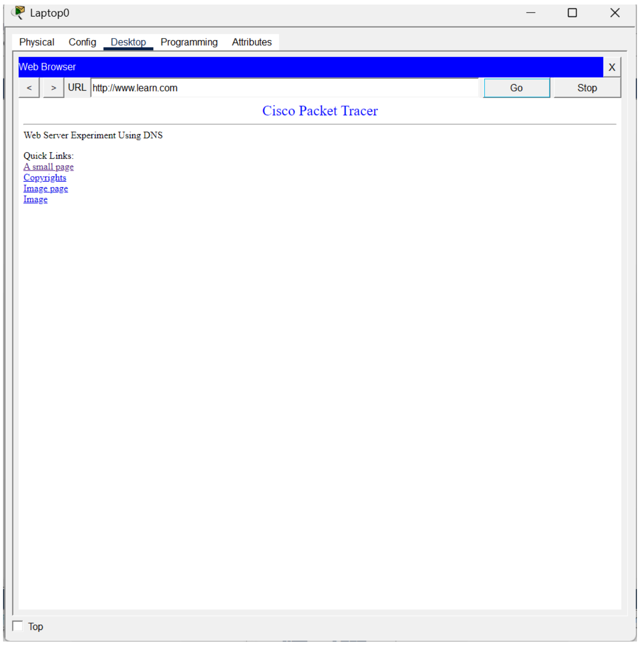
- ping 192.168.1.5 → tests DNS server connection
- ping 192.168.1.6 → tests web server connection

Open **Desktop** > **Web Browser** and enter:
http://www.letslearn.com
The webpage should load successfully.

6. Extra Tasks

- Modify the HTML page and verify changes appear in browser
- Add more PCs and configure their IP, gateway, and DNS similarly

Output:



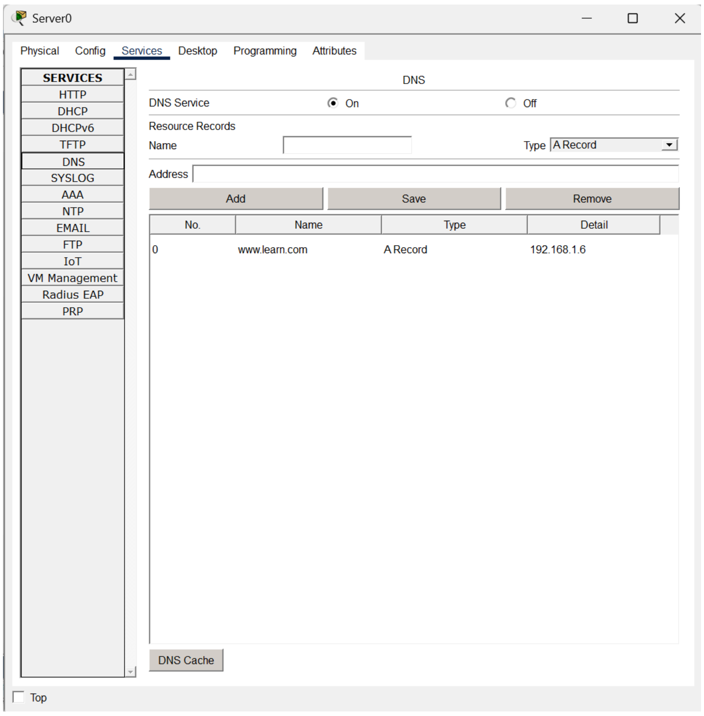
URL http://www.learn.com

Cisco Packet Tracer

Web Server Experiment Using DNS

Quick Links:
A small page
Copyrights
Image page
Image

Fig 3.1 Browser search output



DNS

DNS Service ● On ○ Off

Resource Records

Name

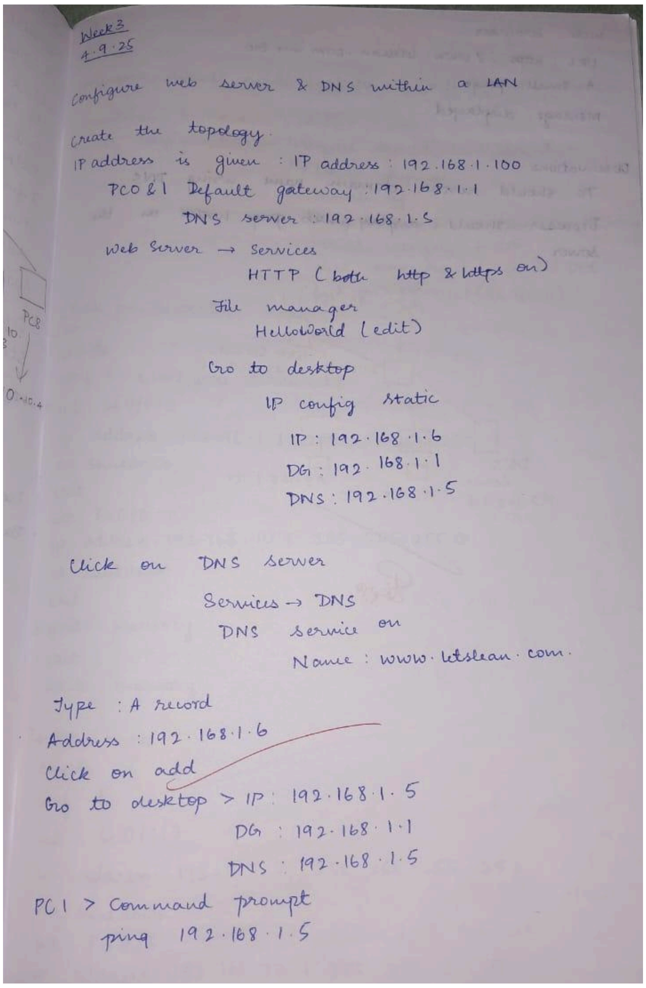
Type A Record

Address

No.	Name	Type	Detail
0	www.learn.com	A Record	192.168.1.6

Fig 3.2 DNS Server Records

Observation:



Week 3
4.9.25

Configure web server & DNS within a LAN

Create the topology.

IP address is given : IP address : 192.168.1.100
PC0 & 1 Default gateway : 192.168.1.1
DNS server : 192.168.1.5

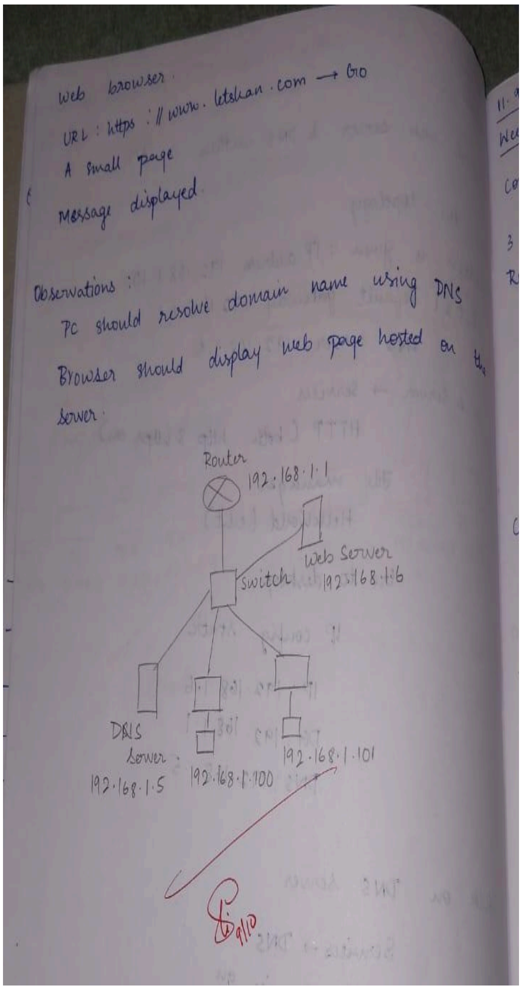
Web Server → Services
HTTP (both http & https on)
File manager
HelloWorld (edit)

Go to desktop
IP config static
IP : 192.168.1.6
DG : 192.168.1.1
DNS : 192.168.1.5

Click on DNS server
Services → DNS
DNS service on
Name : www.letslean.com.

Type : A record
Address : 192.168.1.6
Click on add
Go to desktop > IP : 192.168.1.5
DG : 192.168.1.1
DNS : 192.168.1.5

PC1 > Command prompt
ping 192.168.1.5



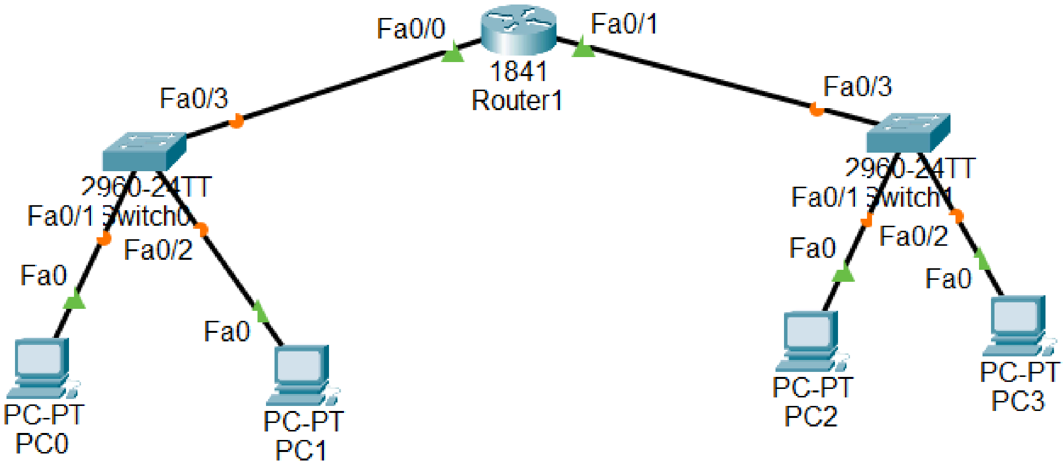
Web browser.
URL : https://www.letshan.com → Go
A small page
Message displayed.

Observations :
PC should resolve domain name using DNS
Browser should display web page hosted on the server.

Program – 4:

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

Topology:



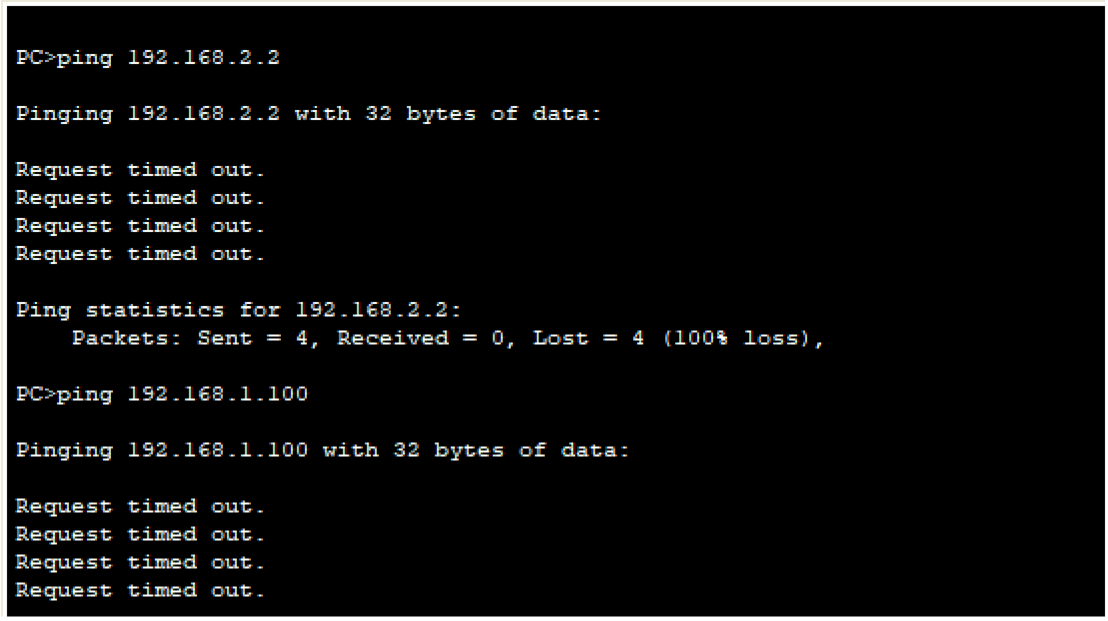
Procedure:

1. Network Setup

Two PCs connected through a switch and router.

- **PC0 IP:** 192.168.2.100
- **PC1 IP:** 192.168.2.101

Output:



```
PC>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

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

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

PC>ping 192.168.1.100

Pinging 192.168.1.100 with 32 bytes of data:

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

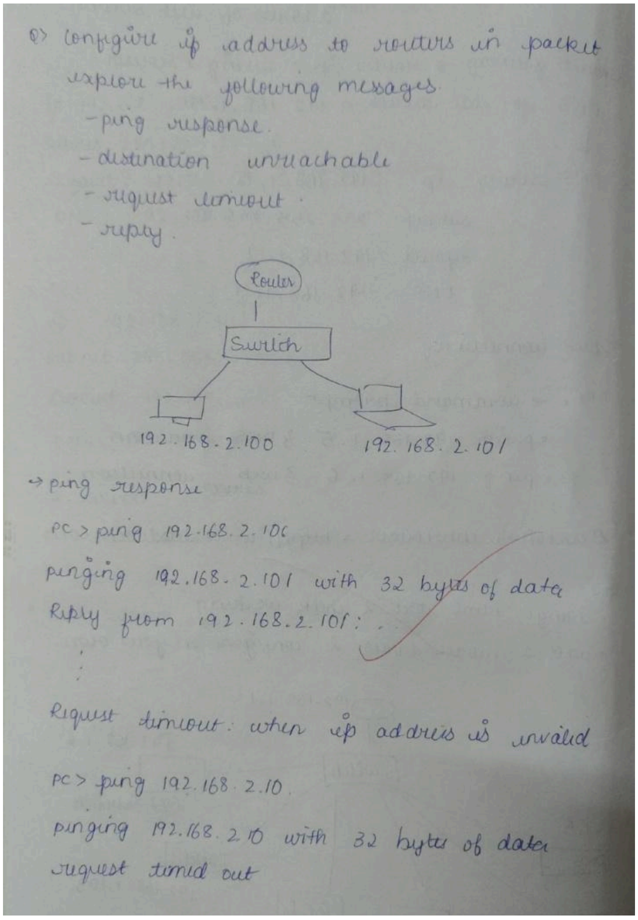
Observation:

```
pc> ping 192.168.2.101
Pinging 192.168.2.101 with 32 bytes of data:
Reply from 192.168.2.101: ...
Reply from 192.168.2.101: ...
Reply from 192.168.2.101: ...
Reply from 192.168.2.101: ...
```

This indicates successful connectivity.

```
pc> ping 192.168.2.10
Pinging 192.168.2.10 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
```

Meaning: No device with this IP is present in the network.



Q> configure ip address to routers in packet explore the following messages.
- ping response.
- destination unreachable
- request timeout.
- reply.

→ ping response

```
pc > ping 192.168.2.101
pinging 192.168.2.101 with 32 bytes of data
Reply from 192.168.2.101: .
:
:
```

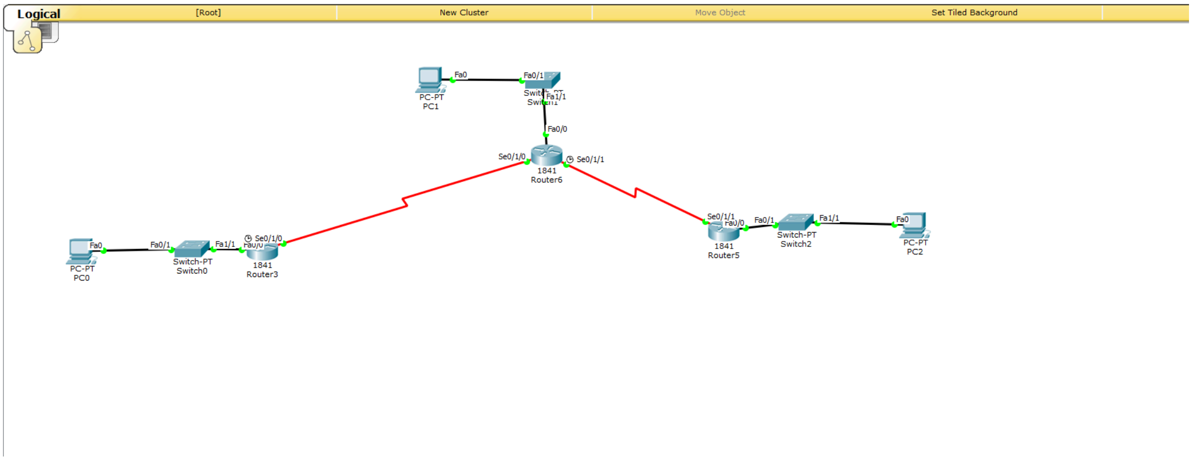
Request timeout: when ip address is invalid

```
pc > ping 192.168.2.10.
pinging 192.168.2.10 with 32 bytes of data
request timed out
```

Program – 5:

Aim: Configure default route, static route to the Router

Topology:



Procedure:

1. Create Topology

Create a 3-router topology where each topology contains:

- 1 PC
- 1 Switch
- 1 Router

2. Router Hardware Setup

For **all 3 routers**:

- Click on **Physical** tab
- Turn **Router OFF**
- Drag & drop **HWIC-2T module**
- Turn **Router ON**

3. Connect the Routers

Use **Serial DCE cable** to interconnect all the 3 routers in serial topology.

Router Configurations:

Router 1

Step 1: Configure Serial Interface

```
Router> enable
Router# configure terminal
Router(config)# int Se0/1/0
Router(config-if)# ip address 172.16.1.1 255.255.255.252
Router(config-if)# no shutdown
Router(config-if)# exit
```

Step 2: Configure FastEthernet

```
Router(config)# interface Fa0/0
Router(config-if)# ip address 192.168.10.1 255.255.255.0
Router(config-if)# no shutdown
Router(config-if)# exit
```

Step 3: Save Configuration

```
Router# write memory
Router# exit
```

Router 2

```
Router> enable
Router# configure terminal
Router(config)# hostname R2
```

Serial Interface (to Router 1)

```
R2(config)# int Se0/1/0
R2(config-if)# ip address 172.16.1.2 255.255.255.252
R2(config-if)# no shutdown
```

FastEthernet Interface

```
R2(config)# int Fa0/0
R2(config-if)# ip address 192.168.20.1 255.255.255.0
R2(config-if)# no shutdown
```

Serial Interface (to Router 3)

```
R2(config)# int Se0/1/1
R2(config-if)# ip address 172.16.2.1 255.255.255.252
R2(config-if)# no shutdown
R2(config-if)# exit
R2# write memory
```

Router 3

```
Router> enable
Router# configure terminal
Router(config)# hostname R3
```

Serial Interface (to Router 2)

```
R3(config)# int Se0/1/1
R3(config-if)# ip address 172.16.2.2 255.255.255.252
R3(config-if)# no shutdown
```

FastEthernet Interface

```
R3(config)# int Fa0/0
R3(config-if)# ip address 192.168.30.1 255.255.255.0
R3(config-if)# no shutdown
R3(config-if)# exit
R3# write memory
```

PC IP Configuration

PC0

- IP: 192.168.10.10
- Default Gateway: 192.168.10.1

PC1

- IP: 192.168.20.10
- Default Gateway: 192.168.20.1

PC2

- IP: 192.168.30.10
- Default Gateway: 192.168.30.1

Static Route Configuration

Router 1

```
R1> enable
R1# configure terminal
R1(config)# hostname R1
R1(config)# ip route 192.168.20.0 255.255.255.0 172.16.1.2
R1(config)# ip route 192.168.30.0 255.255.255.0 172.16.1.2
R1# write memory
```

Router 2

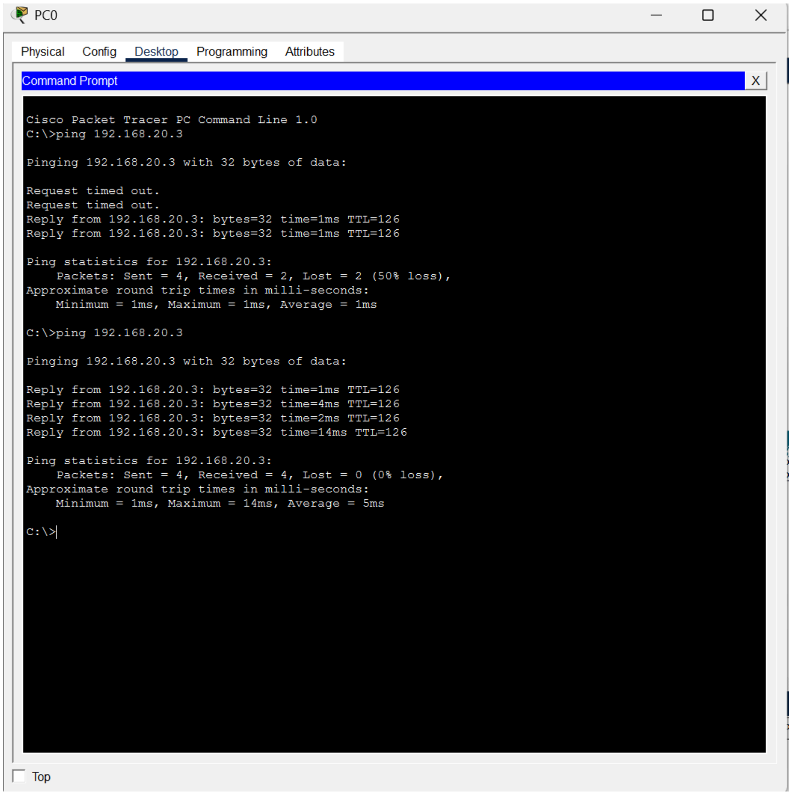
```
R2> enable
```

```
R2# configure terminal
R2(config)# ip route 192.168.10.0 255.255.255.0 172.16.1.1
R2(config)# ip route 192.168.30.0 255.255.255.0 172.16.2.2
R2# write memory
```

Router 3 (Default Route)

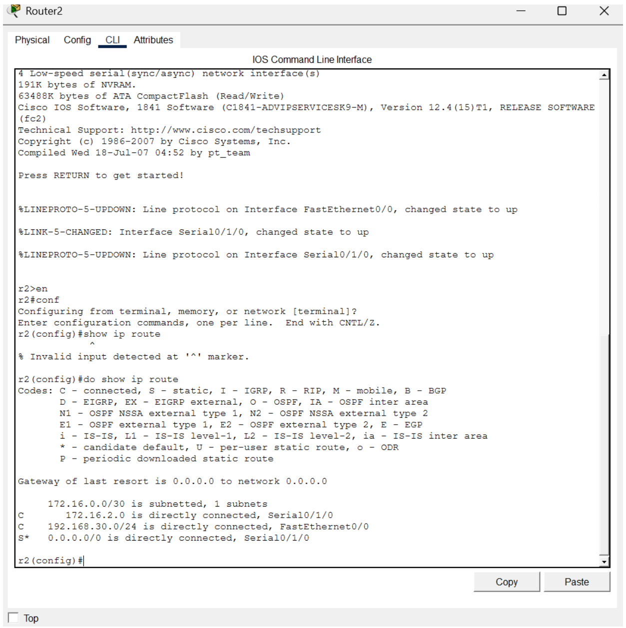
```
R3> enable
R3# configure terminal
R3(config)# ip route 0.0.0.0 0.0.0.0 Se0/1/1
R3# write memory
```

Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data:

Request timed out.
Request timed out.
Reply from 192.168.20.3: bytes=32 time=1ms TTL=126
Reply from 192.168.20.3: bytes=32 time=1ms TTL=126

Ping statistics for 192.168.20.3:
    Packets: Sent = 4, Received = 2, Lost = 2 (50% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms

C:\>ping 192.168.20.3

Pinging 192.168.20.3 with 32 bytes of data:

Reply from 192.168.20.3: bytes=32 time=1ms TTL=126
Reply from 192.168.20.3: bytes=32 time=4ms TTL=126
Reply from 192.168.20.3: bytes=32 time=2ms TTL=126
Reply from 192.168.20.3: bytes=32 time=14ms TTL=126

Ping statistics for 192.168.20.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 14ms, Average = 5ms

C:\>
```

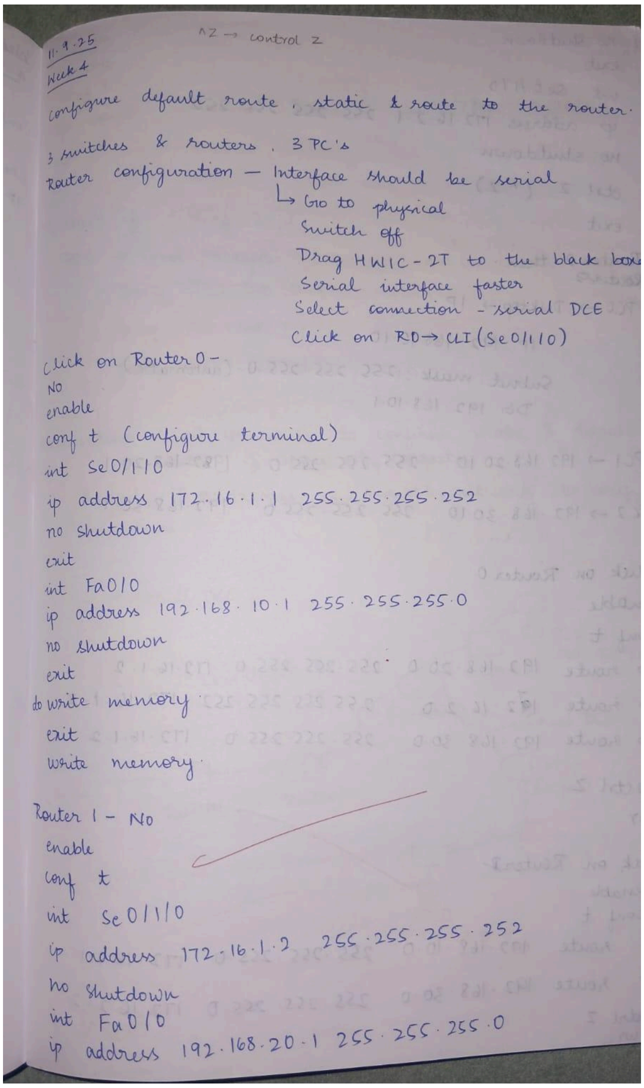
Fig 5.1 ping to PC1 from PC0



```
4 Low-speed serial(sync/async) network interface(s)
191K bytes of NVRAM.
63488K bytes of ATA CompactFlash (Read/Write)
Cisco IOS Software, 1841 Software (C1841-ADVIPSERVICESK9-M), Version 12.4(15)T1, RELEASE SOFTWARE
(fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2007 by Cisco Systems, Inc.
Compiled Wed 18-Jul-07 04:52 by pt_team

Press RETURN to get started!


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

%LINK-5-CHANGED: Interface Serial0/1/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/1/0, changed state to up


r2>en
r2#conf
Configuring from terminal, memory, or network [terminal]?
Enter configuration commands, one per line.  End with CNTL/Z.
r2(config)#show ip route
            ^
% Invalid input detected at '^' marker.

r2(config)#do 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 0.0.0.0 to network 0.0.0.0

     172.16.0.0/30 is subnetted, 1 subnets
C       172.16.2.0 is directly connected, Serial0/1/0
C    192.168.30.0/24 is directly connected, FastEthernet0/0
S*   0.0.0.0/0 is directly connected, Serial0/1/0

r2(config)#
```

Fig 5.2 ip route information in default router

Observation:



11.9.25
Week 4

^Z → control z

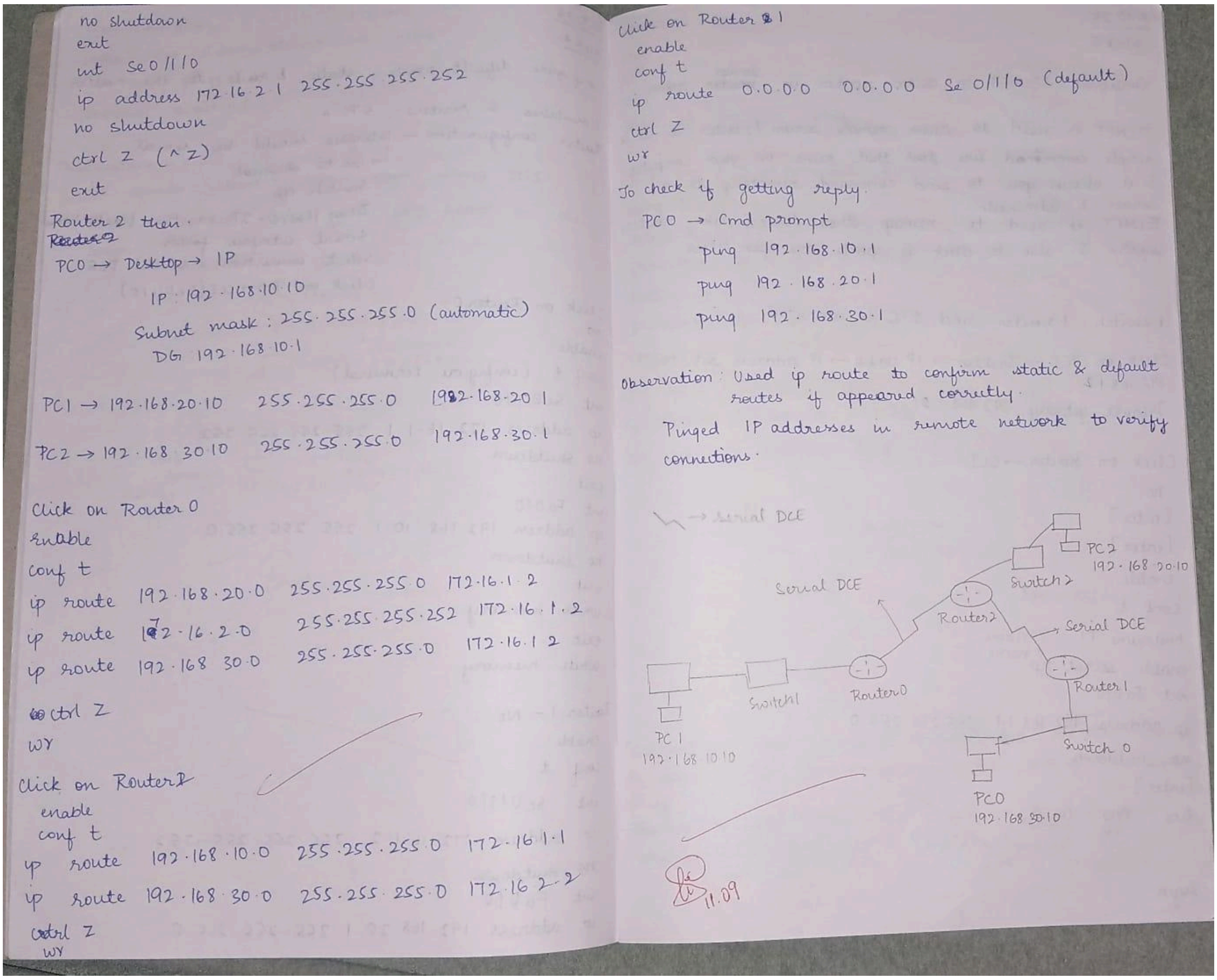
configure default route, static & route to the router.

3 switches & routers, 3 PC's

Router configuration — Interface should be serial
↳ Go to physical
Switch off
Drag HWIC-2T to the black bo
Serial interface faster
Select connection - serial DCE
Click on R0→ CLI (Se 0/1/0)

Click on Router 0 -
No
enable
conf t (configure terminal)
int Se 0/1/0
ip address 172.16.1.1 255.255.255.252
no shutdown
exit
int Fa0/0
ip address 192.168.10.1 255.255.255.0
no shutdown
exit
do write memory
exit
write memory.

Router 1 - No
enable
conf t
int Se 0/1/0
ip address 172.16.1.2 255.255.255.252
no shutdown
int Fa 0/0
ip address 192.168.20.1 255.255.255.0



```
no shutdaon
exit
int Se0/1/0
ip address 172.16.2.1 255.255.255.252
no shutdown
ctrl Z (^Z)
exit
```

Router 2 then.
~~Router2~~

PC0 → Desktop → IP

IP: 192.168.10.10
Subnet mask: 255.255.255.0 (automatic)
DG 192.168.10.1

PC1 → 192.168.20.10 255.255.255.0 19~~8~~2.168.20.1

PC2 → 192.168.30.10 255.255.255.0 192.168.30.1

Click on Router 0

```
enable
conf t
ip route 192.168.20.0 255.255.255.0 172.16.1.2
ip route 1~~9~~72.16.2.0 255.255.255.252 172.16.1.2
ip route 192.168.30.0 255.255.255.0 172.16.1.2
ctrl Z
wr
```

Click on Router~~2~~

```
enable
conf t
ip route 192.168.10.0 255.255.255.0 172.16.1.1
ip route 192.168.30.0 255.255.255.0 172.16.2.2
ctrl Z
wr
```

Click on Router ~~2~~ 1

```
enable
conf t
ip route 0.0.0.0 0.0.0.0 Se 0/1/0 (default)
ctrl Z
wr
```

To check if getting reply.

PC0 → Cmd prompt

```
ping 192.168.10.1
ping 192.168.20.1
ping 192.168.30.1
```

Observation: Used ip route to confirm static & default routes if appeared correctly.

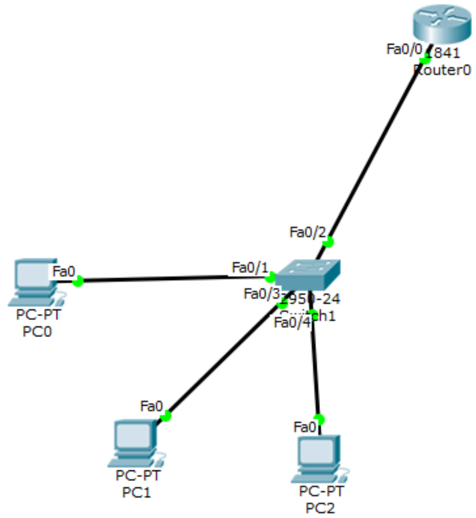
Pinged IP addresses in remote network to verify connections.

11.09

Program – 6:

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

Topology:



Procedure:

1. Topology

- **PC0 → Switch0 → Router0**

2. PC0 IP Configuration

- **IP Address:** 192.168.1.2
- **Default Gateway:** 192.168.1.1

3. Configure Router for Telnet

Open **CLI** on Router0:

```
Router> enable
Router# configure terminal
Router(config)# hostname R1
```

Assign IP Address to Router Interface

```
R1(config)# interface Fa0/0
R1(config-if)# ip address 192.168.1.1 255.255.255.0
R1(config-if)# no shutdown
R1(config-if)# exit
```

4. Configure Telnet (VTY lines)

```
R1(config)# line vty 0 5
R1(config-line)# login
R1(config-line)# password cpl
R1(config-line)# exit
```

5. Verify Interface Status

```
R1# show ip interface brief
```

6. Test Telnet Connection from PC0

Open **Command Prompt** on PC0:

Ping Test

ping 192.168.1.1
(Output: success)

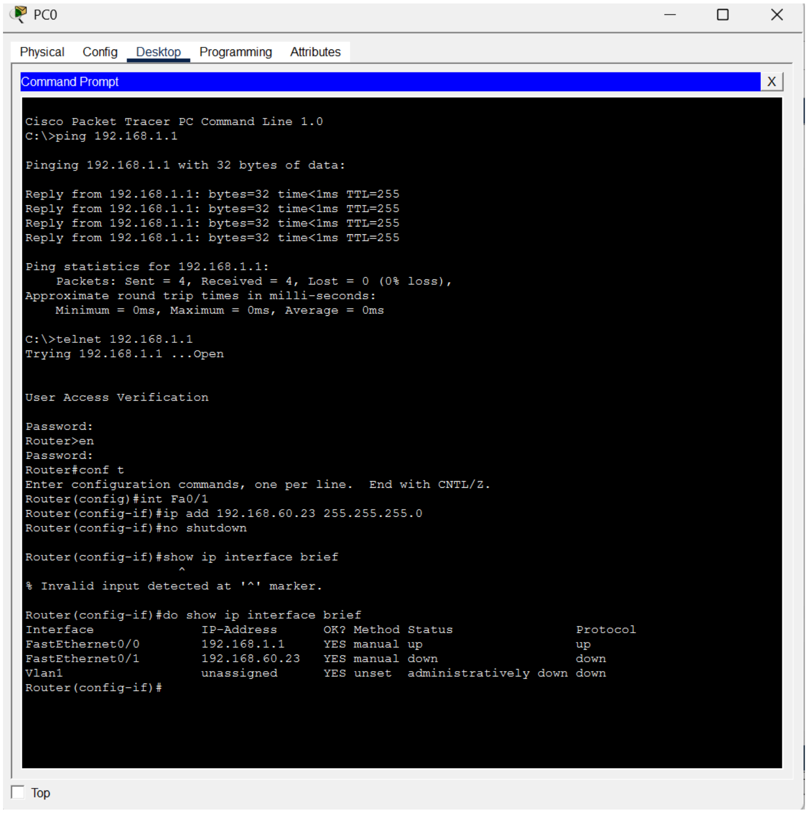
Telnet Access

telnet 192.168.1.1
password: cpl

7. Further Configuration (Second Interface)

R1> enable
Password: cpl
R1# configure terminal
R1(config)# interface Fa0/1
R1(config-if)# ip address 192.168.1.4 255.255.255.0
R1(config-if)# exit
R1# show ip interface brief

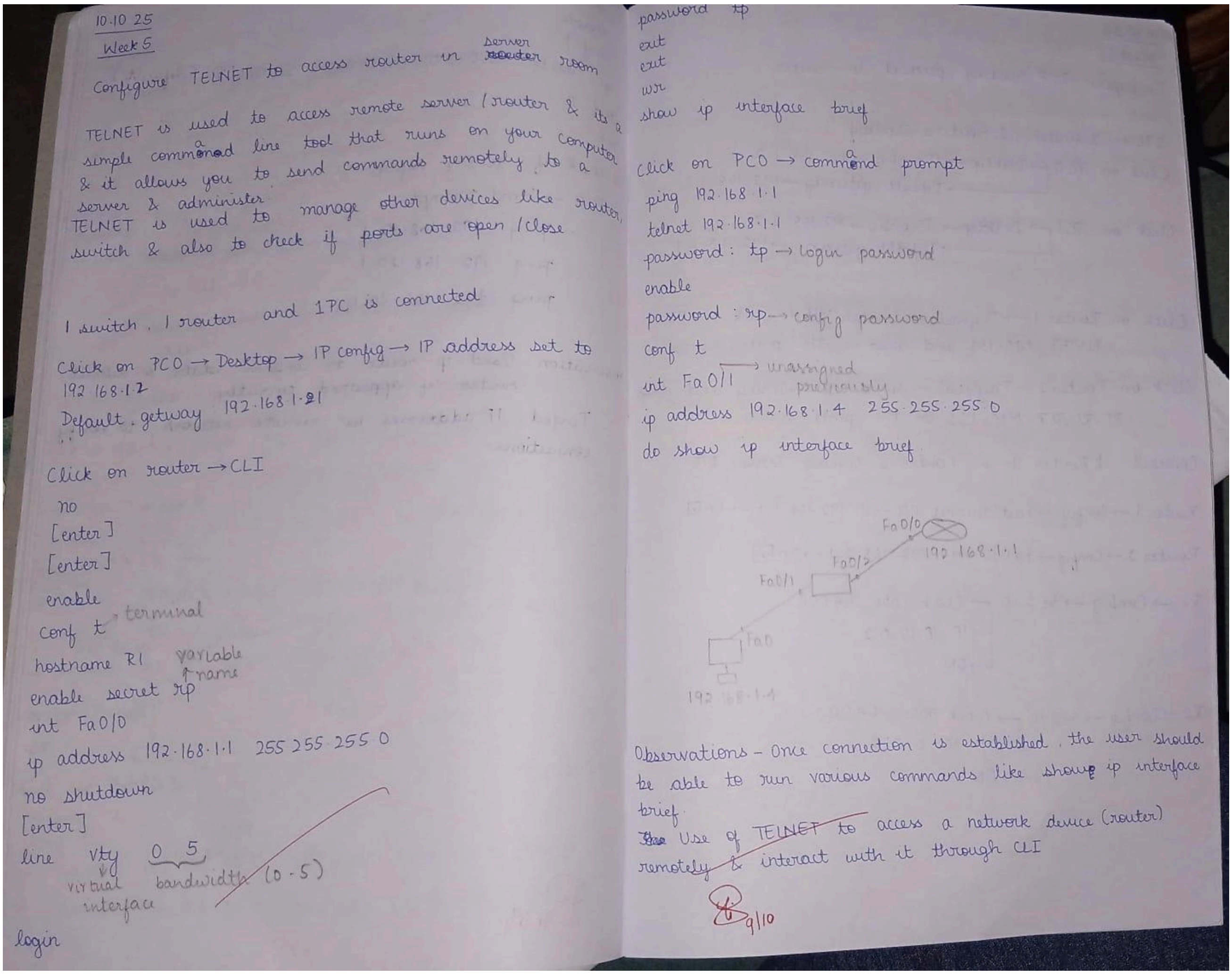
Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.1.1

Pinging 192.168.1.1 with 32 bytes of data:

Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255

Ping statistics for 192.168.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>telnet 192.168.1.1
Trying 192.168.1.1 ...Open


User Access Verification

Password:
Router>en
Password:
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int Fa0/1
Router(config-if)#ip add 192.168.60.23 255.255.255.0
Router(config-if)#no shutdown

Router(config-if)#show ip interface brief
                    ^
% Invalid input detected at '^' marker.

Router(config-if)#do show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        192.168.1.1     YES manual up                    up
FastEthernet0/1        192.168.60.23   YES manual down                  down
Vlan1                  unassigned      YES unset  administratively down down
Router(config-if)#
```

Fig 6.1 – Remote access from PC0 to router

Observation:



10.10 25

Week 5

Configure TELNET to access router in ~~router~~ server room

TELNET is used to access remote server / router & its a simple command line tool that runs on your computer & it allows you to send commands remotely to a server & administer TELNET is used to manage other devices like router, switch & also to check if ports are open / close

1 switch, 1 router and 1PC is connected

Click on PC0 → Desktop → IP config → IP address set to 192 168.1.2

Default. getway 192.168 1.21

Click on router → CLI

no

[enter]

[enter]

enable

conf t → terminal

hostname R1 variable name

enable secret rp

int Fa0/0

ip address 192.168.1.1 255 255.255.0

no shutdown

[enter]

line vty $\underbrace{0\ \ 5}_{\text{bandwidth}}$ (0 - 5)

vty ↓ virtual interface

login

password tp

exit

exit

wr

show ip interface brief

Click on PC0 → command prompt

ping 192.168 1.1

telnet 192.168.1.1

password : tp → login password

enable

password : rp → config password

conf t

int Fa0/1 → unassigned previously

ip address 192.168.1.4 255.255.255.0

do show ip interface brief

Observations - Once connection is established, the user should be able to run various commands like show ip interface brief.

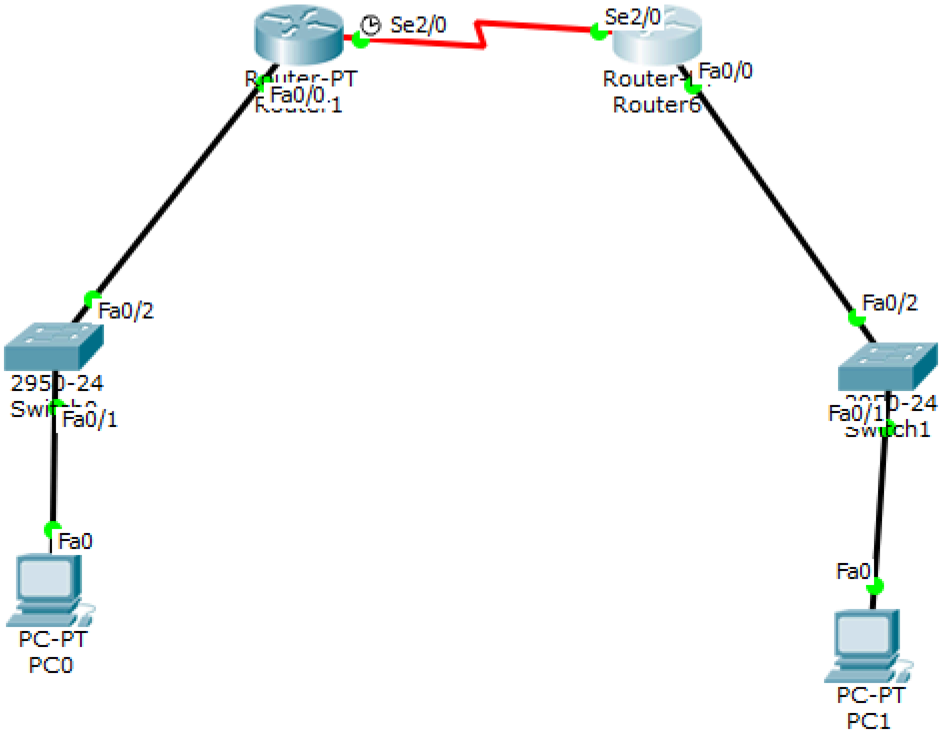
Use of TELNET to access a network device (router) remotely & interact with it through CLI

9/10

Program – 7:

Aim: Configure RIP routing Protocol in Routers.

Topology:



Procedure:

1. Topology

- Two routers connected using Serial DCE.
- Router0 ↔ Router1.
- Router0 connected to Switch0 → PC0.
- Router1 connected to Switch1 → PC1.

2. PC Configuration

- **PC0**
  - IP Address: 192.168.1.2
  - Default Gateway: 192.168.1.1
- **PC1**
  - IP Address: 192.168.2.2
  - Default Gateway: 192.168.2.1

3. Router0 Configuration

- Go to CLI:
  - enable
  - configure terminal
- Configure FastEthernet0/0:
  - interface Fa0/0
  - ip address 192.168.1.1 255.255.255.0
  - no shutdown
  - exit
- Configure Serial Interface:
  - interface Se0/2
  - clock rate 64000
  - ip address 10.10.0.2 255.255.255.252

  - no shutdown
  - exit
- Configure RIP:
  - router rip
  - network 192.168.1.0
  - network 10.0.0.0
  - exit
- Save:
  - write memory

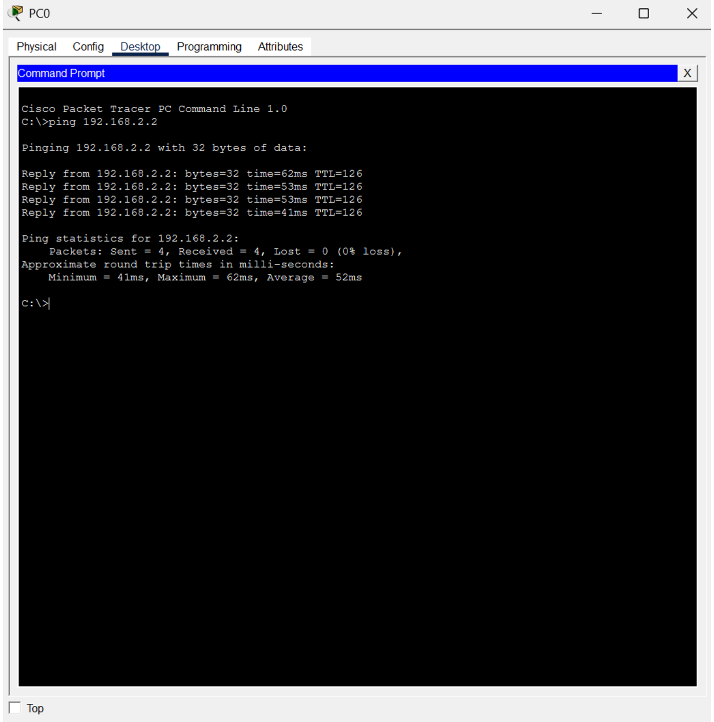
4. Router1 Configuration

- Go to CLI:
  - enable
  - configure terminal
- Configure FastEthernet0/0:
  - interface Fa0/0
  - ip address 192.168.2.1 255.255.255.0
  - no shutdown
  - exit
- Configure Serial Interface:
  - interface Se0/2
  - clock rate 64000
  - ip address 10.0.0.3 255.255.255.252
  - no shutdown
  - exit
- Configure RIP:
  - router rip
  - network 192.168.2.0
  - network 10.0.0.0
  - exit
- Save:
  - write memory

5. RIP Networks Used

- Router0: 192.168.1.0 and 10.0.0.0
- Router1: 192.168.2.0 and 10.0.0.0

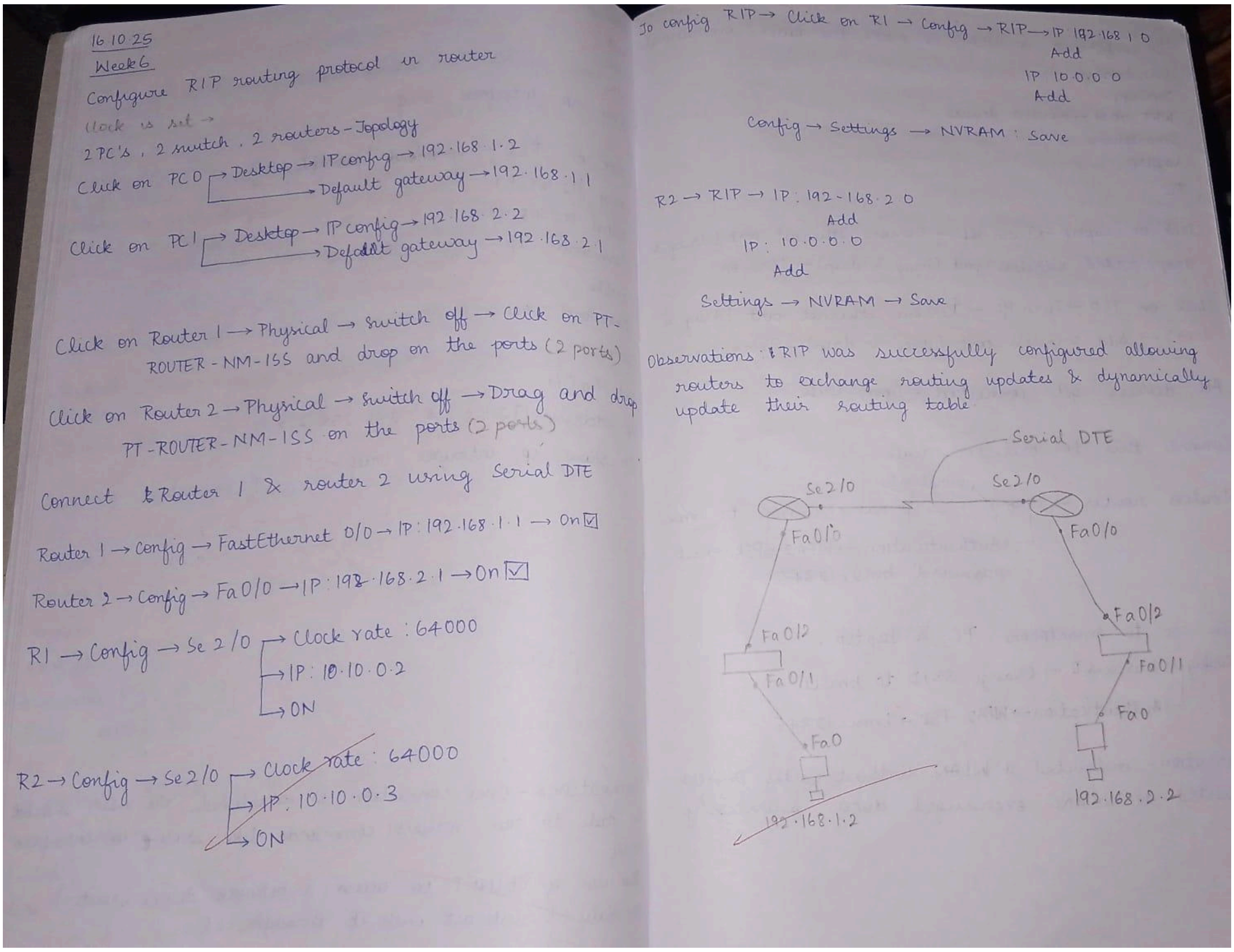
Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Reply from 192.168.2.2: bytes=32 time=62ms TTL=126
Reply from 192.168.2.2: bytes=32 time=53ms TTL=126
Reply from 192.168.2.2: bytes=32 time=53ms TTL=126
Reply from 192.168.2.2: bytes=32 time=41ms TTL=126

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 41ms, Maximum = 62ms, Average = 52ms

C:\>
```

Fig 7.1 Ping PC0 to PC1

Observation:



16.10.25

Week 6

Configure RIP routing protocol in router

Clock is set →

2 PC's, 2 switch, 2 routers - Topology

Click on PC 0 → Desktop → IP config → 192.168.1.2
→ Default gateway → 192.168.1.1

Click on PC 1 → Desktop → IP config → 192.168.2.2
→ Default gateway → 192.168.2.1

Click on Router 1 → Physical → switch off → Click on PT-ROUTER-NM-1SS and drop on the ports (2 ports)

Click on Router 2 → Physical → switch off → Drag and drop PT-ROUTER-NM-1SS on the ports (2 ports)

Connect ~~&~~ Router 1 & router 2 using Serial DTE

Router 1 → Config → FastEthernet 0/0 → IP : 192.168.1.1 → On ☑

Router 2 → Config → Fa 0/0 → IP : 192.168.2.1 → On ☑

R1 → Config → Se 2/0
→ Clock rate : 64000
→ IP : 10.10.0.2
→ ON

R2 → Config → Se 2/0
→ Clock rate : 64000
→ IP : 10.10.0.3
→ ON

To config RIP → Click on R1 → Config → RIP → IP 192.168.1.0
Add
IP 10.0.0.0
Add

Config → Settings → NVRAM : Save

R2 → RIP → IP : 192.168.2.0
Add
IP : 10.0.0.0
Add

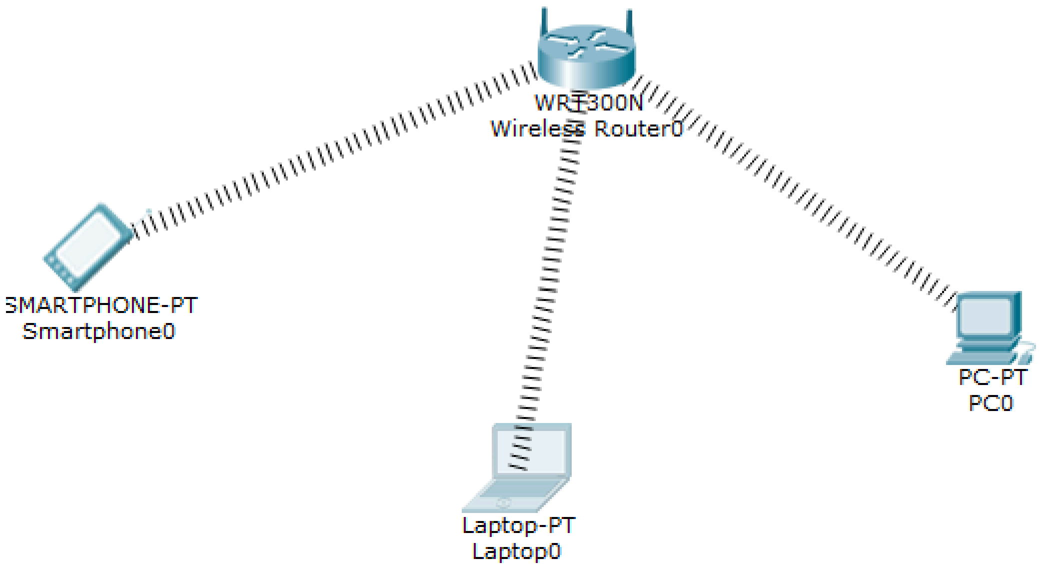
Settings → NVRAM → Save

Observations : RIP was successfully configured allowing routers to exchange routing updates & dynamically update their routing table

Program – 8:

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

Topology:



Procedure:

1. Topology

- Wireless Router0.
- Smartphone0, Laptop0, and PC0 connected wirelessly.

2. Laptop Configuration

- Go to **Physical Tab**.
- Turn off the device.
- Remove Ethernet module.
- Add WiFi module.
- Turn device ON.

3. PC Configuration

- Go to **Physical Tab**.
- Turn off device.
- Remove Ethernet.
- Add WiFi module.
- Turn ON.

4. Wireless Router Configuration

- Go to **Config → Wireless**.
- Set **SSID: BMSCE**.
- Set **Authentication: WPA2-PSK**.
- Set **Password: BMSCE12345**.

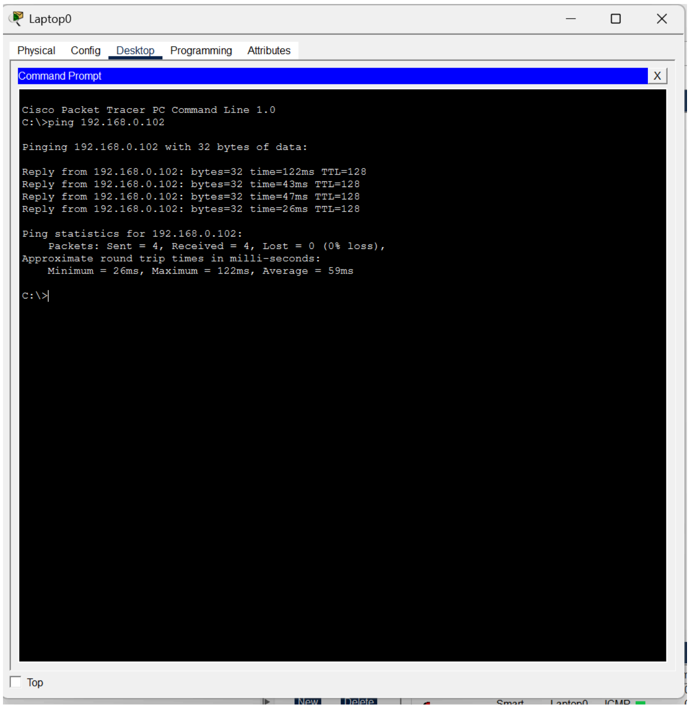
5. Smartphone Configuration

- Enable WiFi.
- Connect to SSID **BMSCE**.
- Enter password **BMSCE12345**.

6. Connect Laptop and PC

- Open WiFi settings → select SSID **BMSCE**.
- Enter password **BMSCE12345**.

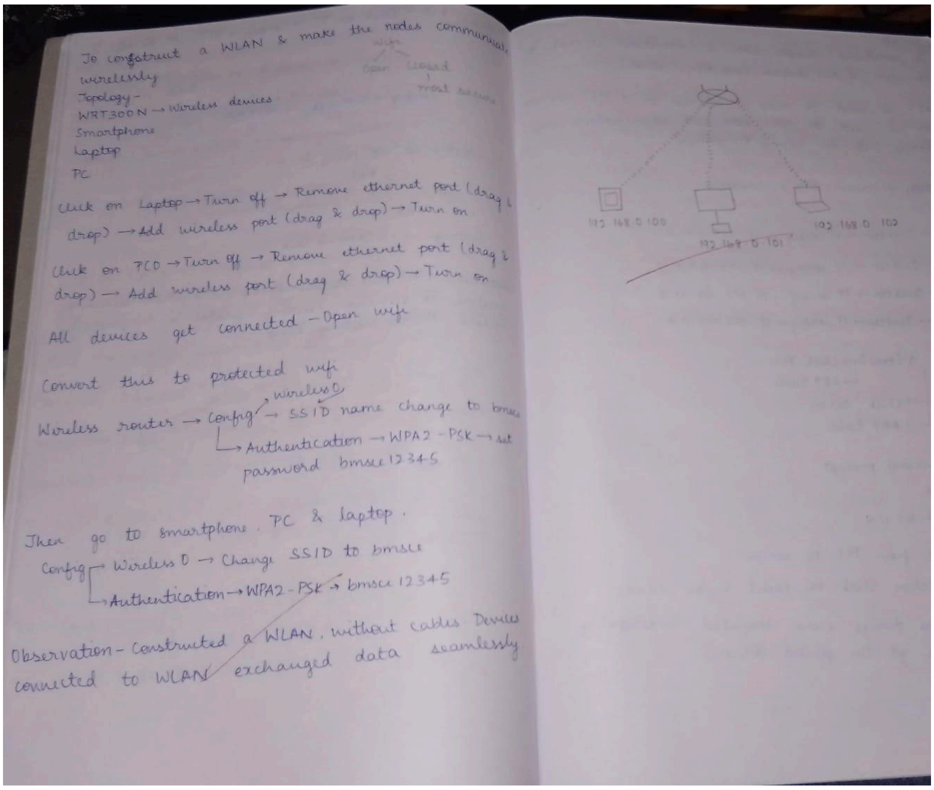
Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.0.102

Pinging 192.168.0.102 with 32 bytes of data:

Reply from 192.168.0.102: bytes=32 time=122ms TTL=128
Reply from 192.168.0.102: bytes=32 time=43ms TTL=128
Reply from 192.168.0.102: bytes=32 time=47ms TTL=128
Reply from 192.168.0.102: bytes=32 time=26ms TTL=128

Ping statistics for 192.168.0.102:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 26ms, Maximum = 122ms, Average = 59ms

C:\>
```

Fig 8.1 data transfer from laptop to mobile

Observation:



To constaruct a WLAN & make the nodes communicate wirelessly

Topology -
WRT300N → Wireless devices
Smartphone
Laptop
PC

Click on Laptop → Turn off → Remove ethernet port (drag & drop) → Add wireless port (drag & drop) → Turn on

Click on PC0 → Turn off → Remove ethernet port (drag & drop) → Add wireless port (drag & drop) → Turn on

All devices get connected - Open wifi

Convert this to protected wifi

Wireless router → Config → wireless0 → SSID name change to bmsce
└→ Authentication → WPA2 - PSK → set password bmsce12345

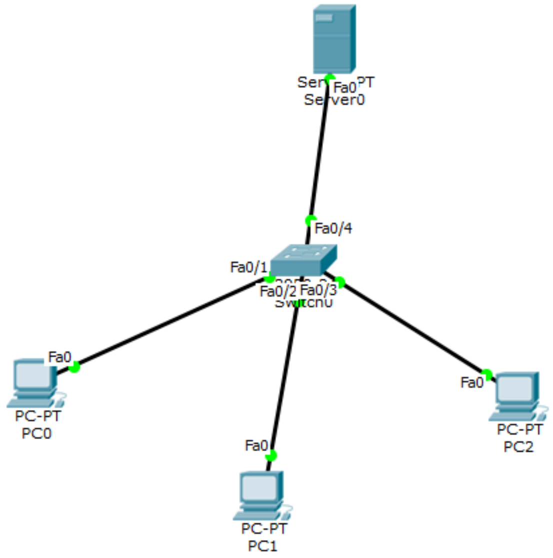
Then go to smartphone, PC & laptop.
Config → Wireless 0 → Change SSID to bmsce
└→ Authentication → WPA2 - PSK → bmsce12345

Observation - Constructed a WLAN, without cables Devices connected to WLAN exchanged data seamlessly

Program – 9:

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

Topology:



Procedure:

1. ARP Concept

- ARP is used to map an IP address to a MAC address.
- ARP works at the data-link layer to resolve MAC address of a destination IP.
- It updates ARP tables with IP–MAC mappings.

2. Topology

- Server0 connected to Switch.
- Switch connected to PC0, PC1, PC2.

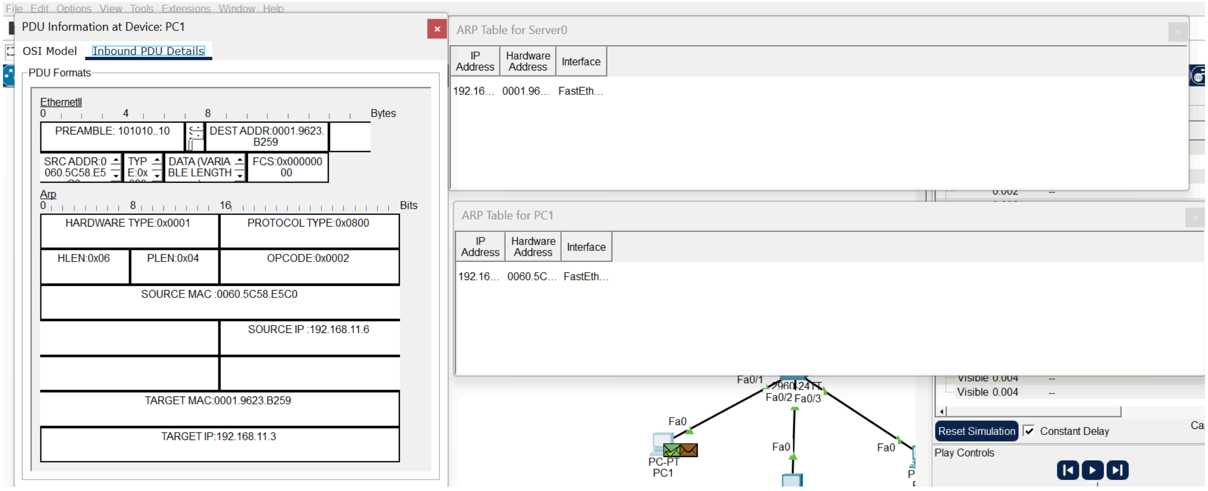
3. IP Configuration

- **PC0** → IP: 192.168.1.1
- **PC1** → IP: 192.168.1.2
- **PC2** → IP: 192.168.1.3
- **Server0** → IP: 192.168.1.4

4. Steps

- Open **Simulation mode** from right side.
- Click on **PC0** or **Server0** → select **ARP Table**.
- Send a packet (simulation) from PC0.
- Click **Outbound PDU Details** → view MAC address.

Output:



PDU Information at Device: PC1

EthernetII

PREAMBLE: 101010..10

DEST ADDR:0001.9623.B259

SRC ADDR:0060.5C58.E5

TYPE:0x

DATA (VARIABLE LENGTH

FCS:0x00000000

Arp

HARDWARE TYPE:0x0001

PROTOCOL TYPE:0x0800

HLEN:0x06

PLEN:0x04

OPCODE:0x0002

SOURCE MAC :0060.5C58.E5C0

SOURCE IP :192.168.11.6

TARGET MAC:0001.9623.B259

TARGET IP:192.168.11.3

ARP Table for Server0

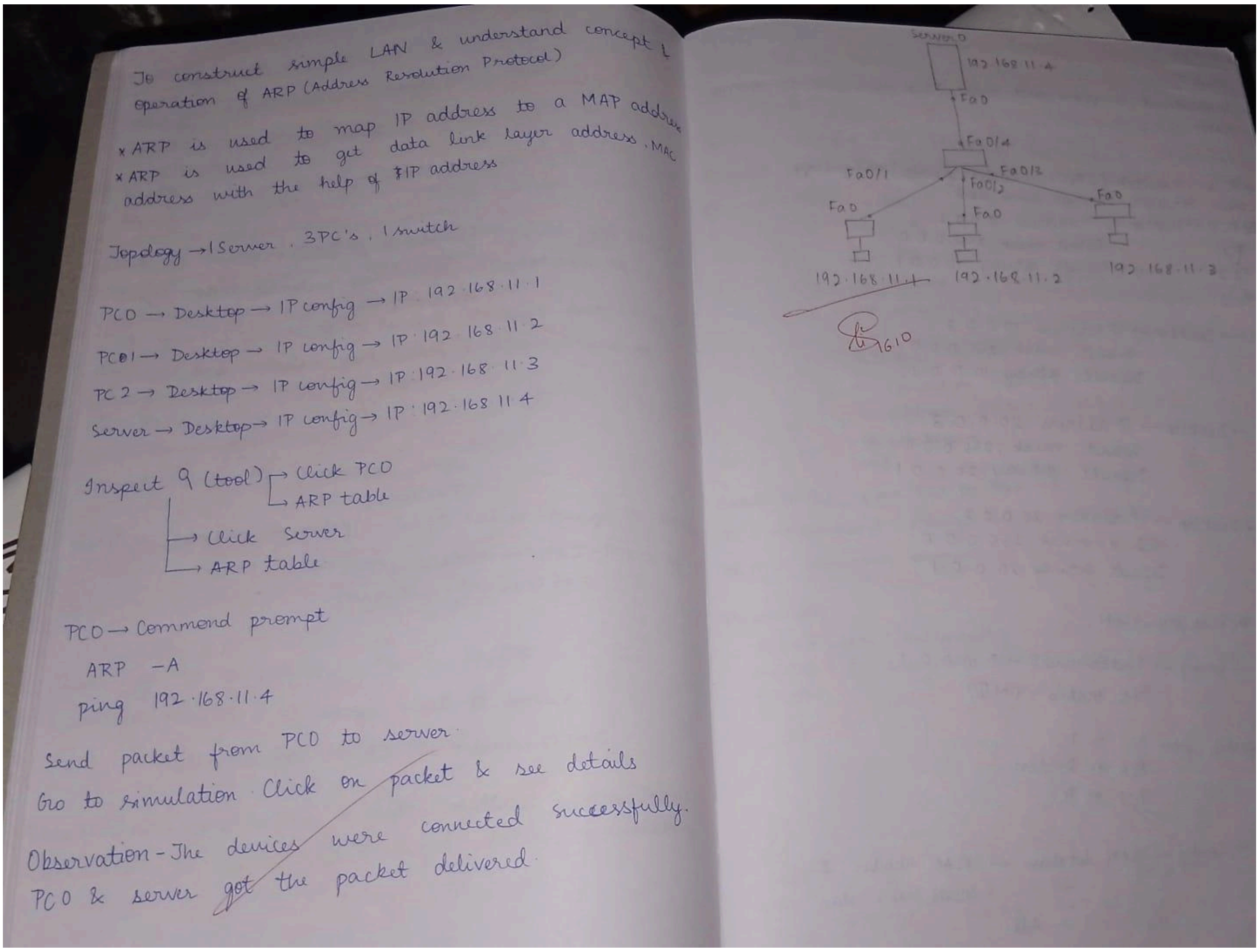
IP Address	Hardware Address	Interface
192.16...	0001.96...	FastEth...

ARP Table for PC1

IP Address	Hardware Address	Interface
192.16...	0060.5C...	FastEth...

Fig 9.1 PDU Information and ARP Table of server and PC0

Observation:



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

× ARP is used to map IP address to a MAP address

× ARP is used to get data link layer address, MAC address with the help of IP address

Topology → 1 Server, 3PC's, 1 switch

PC0 → Desktop → IP config → IP 192.168.11.1

PC01 → Desktop → IP config → IP 192.168.11.2

PC 2 → Desktop → IP config → IP 192.168.11.3

Server → Desktop → IP config → IP 192.168.11.4

Inspect 9 (tool) → Click PC0 → ARP table

→ Click Server → ARP table

PC0 → Commend prompt

ARP -A

ping 192.168.11.4

Send packet from PC0 to server

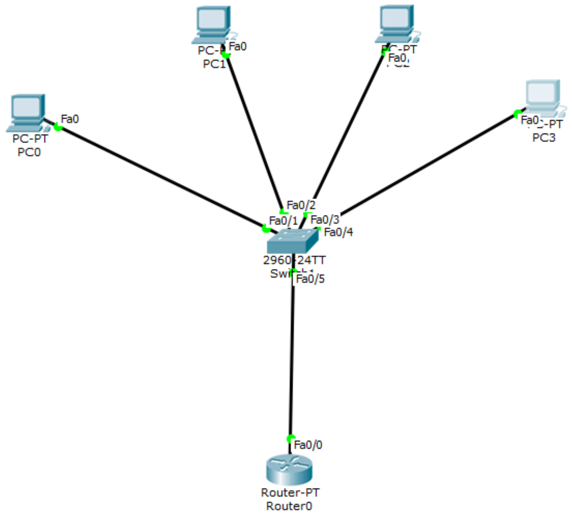
Go to simulation. Click on packet & see details

Observation - The devices were connected successfully. PC0 & server got the packet delivered.

Program – 10:

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

Topology:



Procedure:

1. Topology

- One Router
- One 2960–24TT Switch
- Four PCs (PC0, PC1, PC2, PC3)
- Connections using Copper Straight-Through cables

2. Router Configuration

Open CLI

- enable
- configure terminal

Configure Interfaces

Fa0/0

- interface Fa0/0
- ip address 10.0.0.1 255.0.0.0
- no shutdown
- exit

Fa0/0.1 (Sub-interface for VLAN 20)

- interface Fa0/0.1
- encapsulation dot1Q 2
- ip address 20.0.0.1 255.0.0.0
- no shutdown
- exit

3. PC Configurations

PC0

- Go to **Config → FastEthernet**
- IP Address: **10.0.0.2**
- Default Gateway: **10.0.0.1**

PC1

- Go to **Config → FastEthernet**
- IP Address: **10.0.0.3**
- Default Gateway: **10.0.0.1**

PC2

- Go to **Config → FastEthernet**
- IP Address: **20.0.0.3**
- Default Gateway: **20.0.0.1**

PC3

- Go to **Config → FastEthernet**
- IP Address: **20.0.0.2**
- Default Gateway: **20.0.0.1**

4. Switch Configuration

Create VLANs

- Go to **Config → VLAN Database**
- VLAN Number: **2**
- VLAN Name: **vlan**

Assign Ports to VLANs

Configure Trunk Port

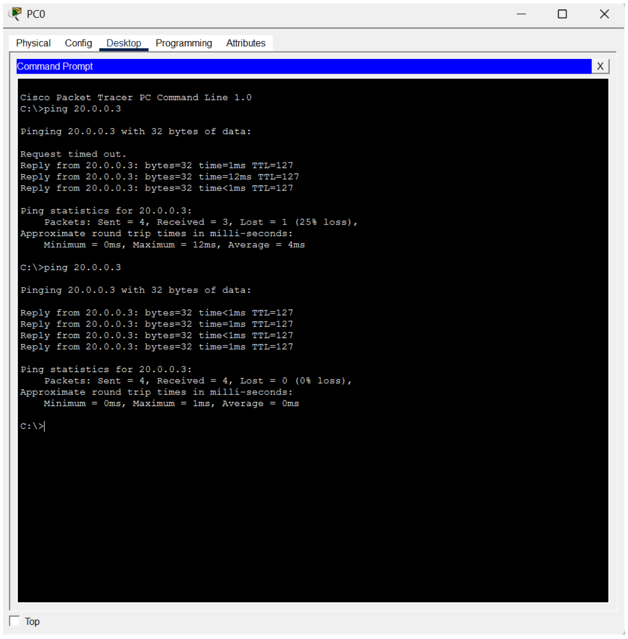
- FastEthernet0/5 (connection to router):
  - Change **Access → Trunk**

Assign VLAN 2 Ports

- Fa0/3 and Fa0/4 (connections to PC2 and PC3):
  - Change VLAN → **2**

(PC0 and PC1 remain in default VLAN 1)

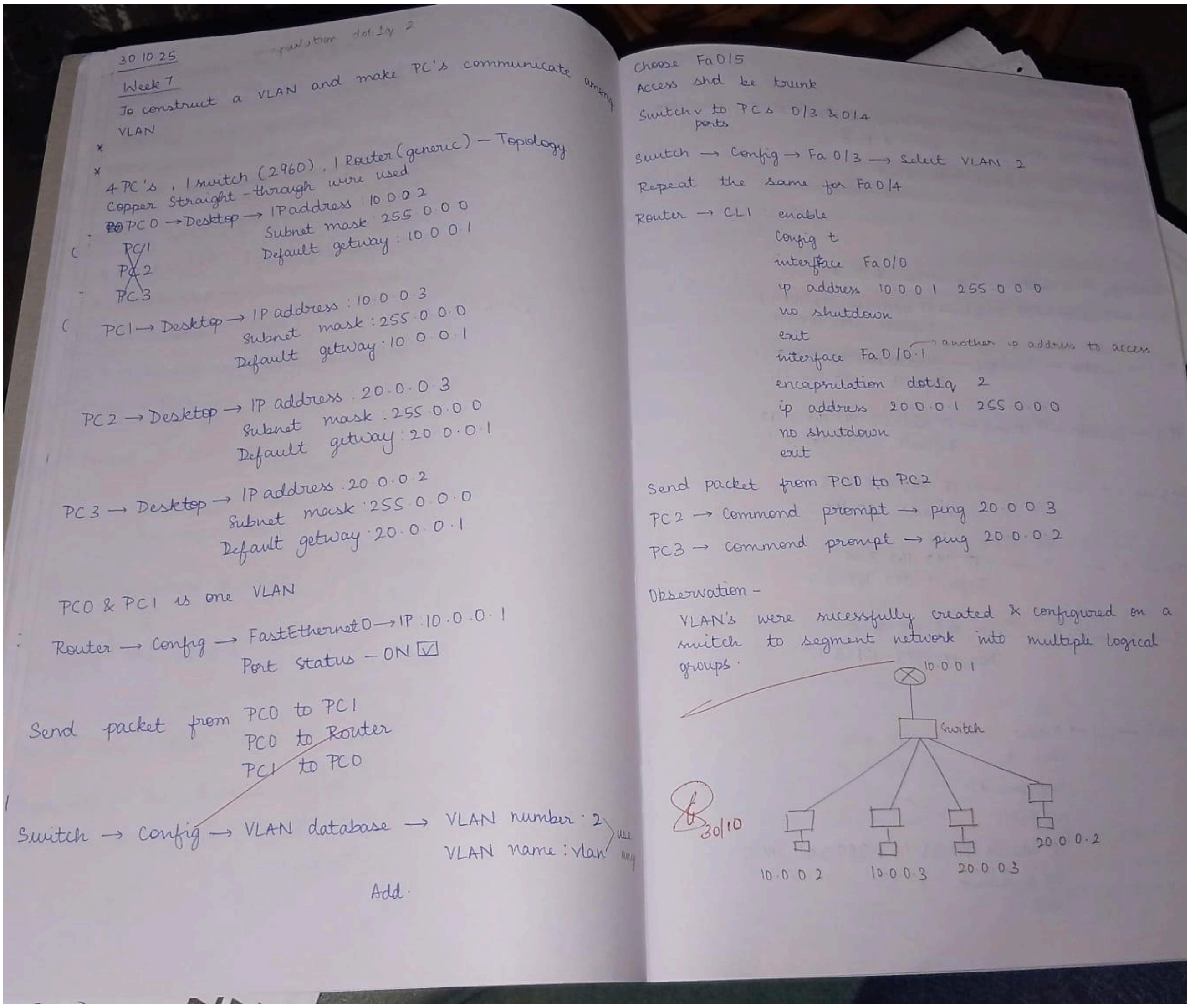
Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 20.0.0.3

Pinging 20.0.0.3 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.3: bytes=32 time=1ms TTL=127
Reply from 20.0.0.3: bytes=32 time=12ms TTL=127
Reply from 20.0.0.3: bytes=32 time<1ms TTL=127

Ping statistics for 20.0.0.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 12ms, Average = 4ms

C:\>ping 20.0.0.3

Pinging 20.0.0.3 with 32 bytes of data:

Reply from 20.0.0.3: bytes=32 time<1ms TTL=127
Reply from 20.0.0.3: bytes=32 time=1ms TTL=127
Reply from 20.0.0.3: bytes=32 time<1ms TTL=127
Reply from 20.0.0.3: bytes=32 time=1ms TTL=127

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

C:\>
```

Fig 10.1 pinging to the other vLan -PC0 to PC2

Observation:



capsulation dot1q 2

30 10 25

Week 7

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

×

×

4 PC's , 1 switch (2960) , 1 Router (generic) – Topology

Copper Straight-through wire used

~~R0~~ PC0 → Desktop → IPaddress 10.0.0.2
Subnet mask 255.0.0.0
Default getway : 10.0.0.1

~~PC1~~
~~PC2~~
~~PC3~~

PC1 → Desktop → IP address : 10.0.0.3
Subnet mask : 255.0.0.0
Default getway : 10.0.0.1

PC2 → Desktop → IP address . 20.0.0.3
Subnet mask . 255.0.0.0
Default getway : 20.0.0.1

PC3 → Desktop → IP address . 20.0.0.2
Subnet mask 255.0.0.0
Default getway . 20.0.0.1

PC0 & PC1 is one VLAN

Router → Config → FastEthernet0 → IP . 10.0.0.1
Port status – ON ☑

Send packet from PC0 to PC1
PC0 to Router
PC1 to PC0

Switch → Config → VLAN database → VLAN number . 2
VLAN name : vlan
use any

Add.

Choose Fa0/5

Access shd be trunk

Switch ports to PCs 0/3 & 0/4

Switch → Config → Fa 0/3 → Select VLAN 2

Repeat the same for Fa 0/4

Router → CLI

```
enable
Config t
interface Fa0/0
ip address 10.0.0.1 255.0.0.0
no shutdown
exit
interface Fa0/0.1    → another ip address to access
encapsulation dot1q 2
ip address 20.0.0.1 255.0.0.0
no shutdown
exit
```

Send packet from PC0 to PC2

PC2 → Commend prompt → ping 20.0.0.3

PC3 → commend prompt → ping 20.0.0.2

Observation –

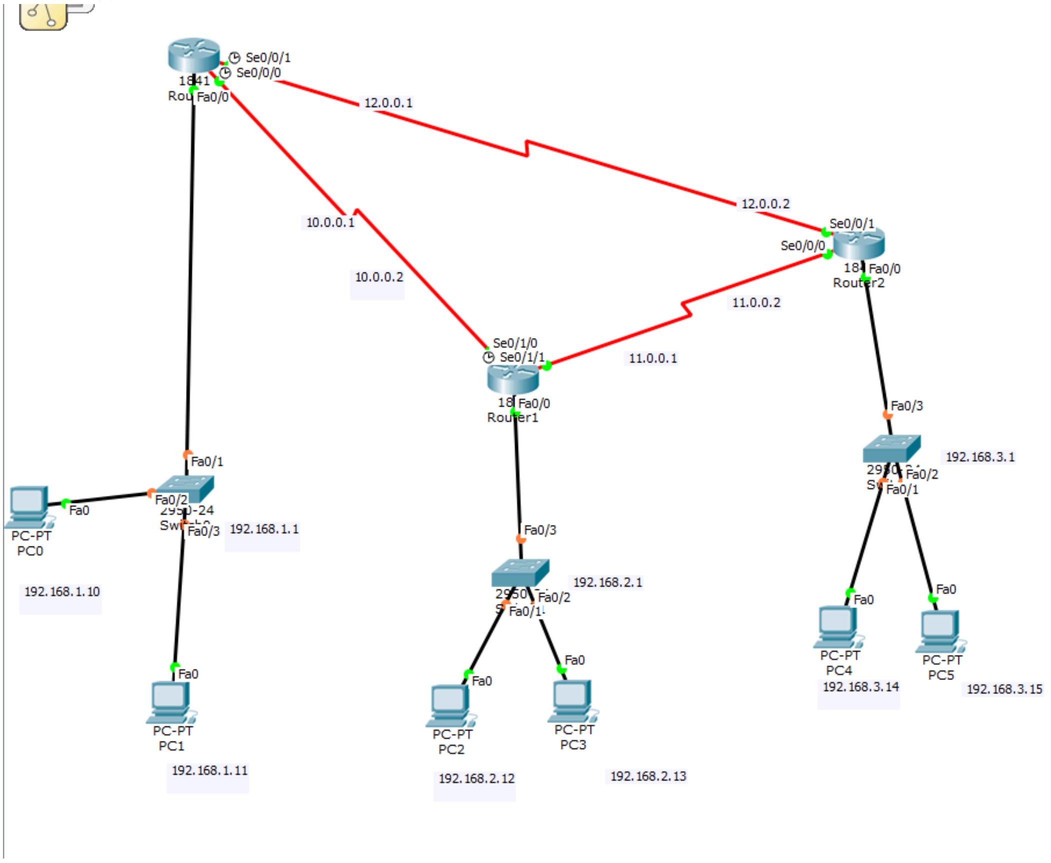
VLAN's were sucessfully created & configured on a switch to segment network into multiple logical groups.

30/10

Program – 11:

Aim: Configure OSPF routing protocol.

Topology:



Procedure:

1. Topology

- Three routers R0, R1, R2 connected using Serial DCE links.
- Each router connected to its own LAN via a switch.
- PCs connected as per diagram:
  - PC0 → 192.168.1.10
  - PC1 → 192.168.1.11
  - PC2 → 192.168.2.12
  - PC3 → 192.168.2.13
  - PC4 → 192.168.3.14
  - PC5 → 192.168.3.15

2. Initial Steps

1. On each router:
   - Go to **Physical tab**
   - Turn OFF the router
   - Insert **HWIC-2T module** (2 times)
   - Turn router ON
2. Create topology using:
   - **Serial DCE** for router-to-router links
   - **Normal copper** for router-to-switch connections
3. Do IP configuration for all PCs as per diagram with switch's IP as default gateway.

3. Router Configuration

Router R0

CLI Steps

- enable
- configure terminal
- hostname R0

Fa0/0

- interface Fa0/0
- ip address 192.168.1.1 255.255.255.0
- no shutdown
- exit

Serial Se0/1/0

- interface Se0/1/0
- ip address 10.0.0.1 255.0.0.0
- clock rate 64000
- no shutdown
- exit

Serial Se0/1/1

- interface Se0/1/1
- ip address 12.0.0.1 255.0.0.0
- clock rate 64000
- no shutdown
- exit

Configure OSPF

- router ospf 1
- network 192.168.1.0 0.0.0.255 area 0
- network 10.0.0.0 0.255.255.255 area 0
- network 12.0.0.0 0.255.255.255 area 0
- exit
- wr

Router R1

CLI Steps

- enable
- configure terminal
- hostname R1

Fa0/0

- interface Fa0/0
- ip address 192.168.2.1 255.255.255.0
- no shutdown
- exit

Se0/1/0

- interface Se0/1/0
- ip address 10.0.0.2 255.0.0.0
- no shutdown
- exit

Se0/1/1

- interface Se0/1/1
- ip address 11.0.0.1 255.0.0.0
- clock rate 64000
- no shutdown
- exit

Configure OSPF

- router ospf 1
- network 192.168.2.0 0.0.0.255 area 0
- network 10.0.0.0 0.255.255.255 area 0

- network 11.0.0.0 0.255.255.255 area 0
- exit
- wr

Router R2
CLI Steps

- enable
- configure terminal
- hostname R2

Fa0/0

- interface Fa0/0
- ip address 192.168.3.1 255.255.255.0
- no shutdown
- exit

Se0/1/0

- interface Se0/1/0
- ip address 11.0.0.2 255.0.0.0
- no shutdown
- exit

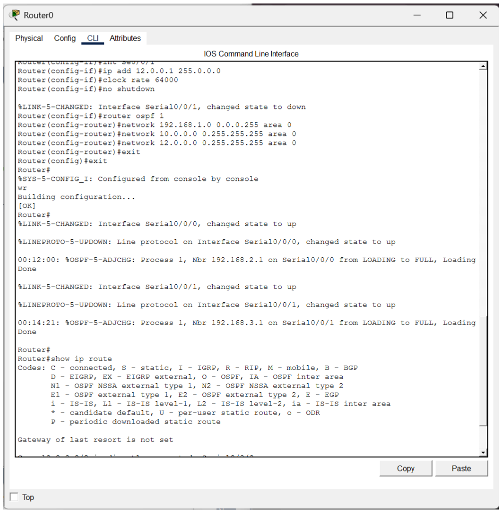
Se0/1/1

- interface Se0/1/1
- ip address 12.0.0.2 255.0.0.0
- no shutdown
- exit

Configure OSPF

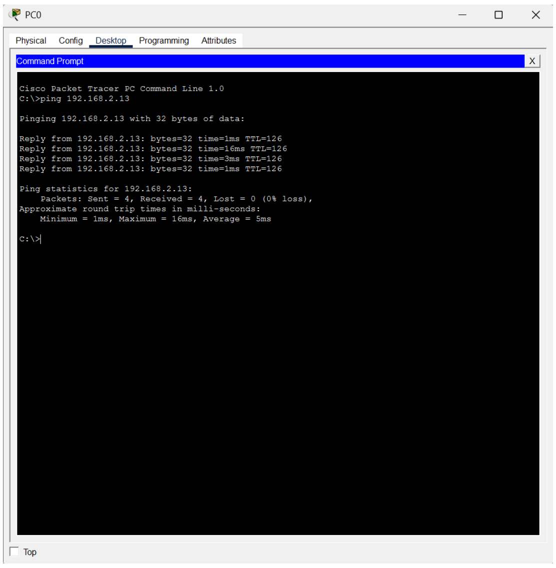
- router ospf 1
- network 192.168.3.0 0.0.0.255 area 0
- network 11.0.0.0 0.255.255.255 area 0
- network 12.0.0.0 0.255.255.255 area 0
- exit
- wr

Output:



```
Router(config-if)#ip add 12.0.0.1 255.0.0.0
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial0/0/1, changed state to down
Router(config-if)#router ospf 1
Router(config-router)#network 192.168.1.0 0.0.0.255 area 0
Router(config-router)#network 10.0.0.0 0.255.255.255 area 0
Router(config-router)#network 12.0.0.0 0.255.255.255 area 0
Router(config-router)#exit
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
wr
Building configuration...
[OK]
Router#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up

00:12:00: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.2.1 on Serial0/0/0 from LOADING to FULL, Loading
Done

%LINK-5-CHANGED: Interface Serial0/0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/1, changed state to up

00:14:21: %OSPF-5-ADJCHG: Process 1, Nbr 192.168.3.1 on Serial0/0/1 from LOADING to FULL, Loading
Done

Router#
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
```

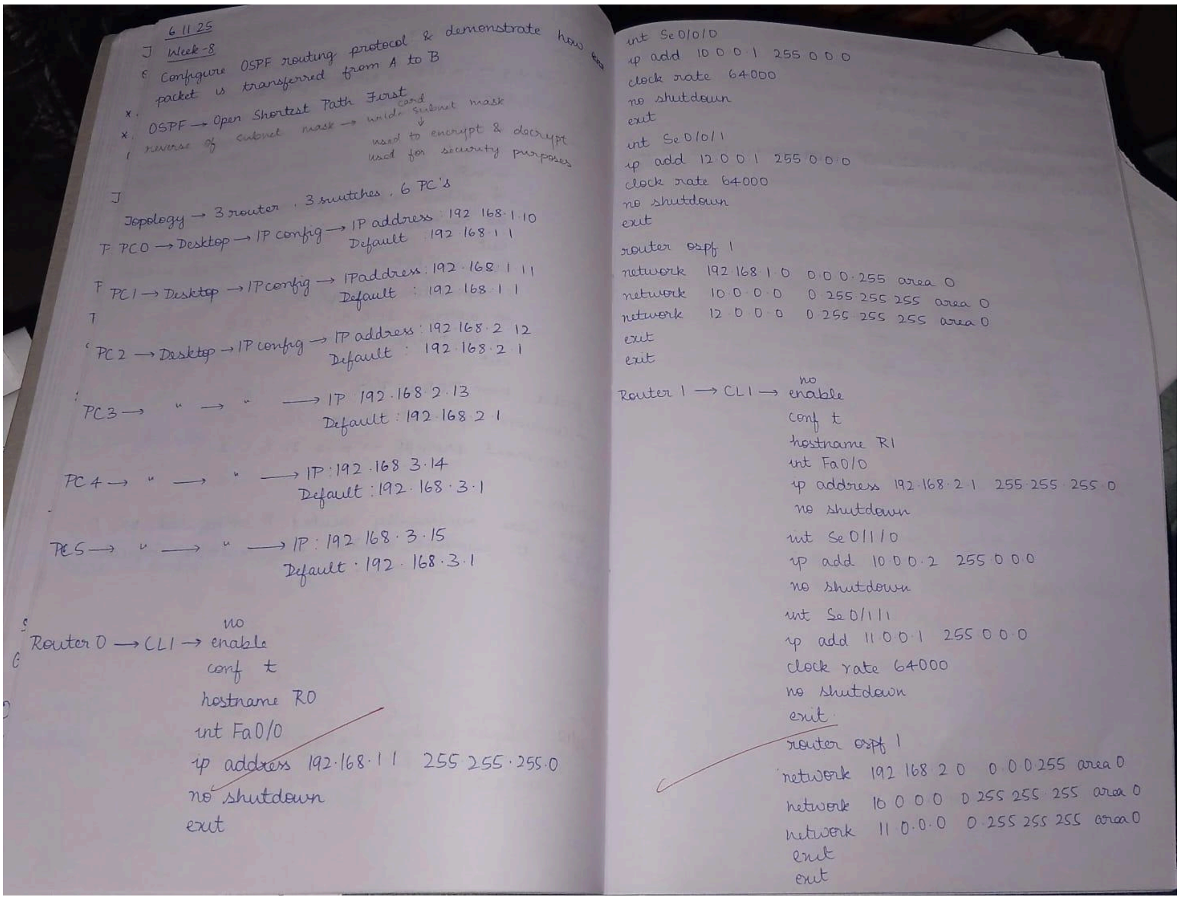
Fig 11.1 OSPF configuration in Router0



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.2.13

Pinging 192.168.2.13 with 32 bytes of data:

Reply from 192.168.2.13: bytes=32 time=1ms TTL=126
Reply from 192.168.2.13: bytes=32 time=16ms TTL=126
Reply from 192.168.2.13: bytes=32 time=3ms TTL=126
Reply from 192.168.2.13: bytes=32 time=1ms TTL=126

Ping statistics for 192.168.2.13:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 16ms, Average = 5ms

C:\>
```

Fig 11.2 ping PC0 to PC3

Observation:



6 11 25
Week-8

Configure OSPF routing protocol & demonstrate how the packet is transferred from A to B

- OSPF → Open Shortest Path First
- reverse of subnet mask → wild card subnet mask
  ↓
  used to encrypt & decrypt
  used for security purposes

Topology → 3 router, 3 switches, 6 PC's

PC0 → Desktop → IP config → IP address 192 168 1 10
Default 192 168 1 1

PC1 → Desktop → IP config → IPaddress : 192 168 1 11
Default : 192 168 1 1

PC2 → Desktop → IP config → IP address : 192 168 2 12
Default : 192 168 2 1

PC3 → " → " → IP 192 168 2 13
Default : 192 168 2 1

PC4 → " → " → IP : 192 168 3 14
Default : 192 168 3 1

PC5 → " → " → IP : 192 168 3 15
Default · 192 168 3 1

Router 0 → CLI → no enable

```
conf t
hostname R0
int Fa0/0
ip address 192.168.1.1 255.255.255.0
no shutdown
exit
int Se0/0/0
ip add 10 0 0 1 255 0 0 0
clock rate 64000
no shutdown
exit
int Se0/0/1
ip add 12 0 0 1 255 0 0 0
clock rate 64000
no shutdown
exit
router ospf 1
network 192 168 1 0 0 0 0 255 area 0
network 10 0 0 0 0 255 255 255 area 0
network 12 0 0 0 0 255 255 255 area 0
exit
exit
```

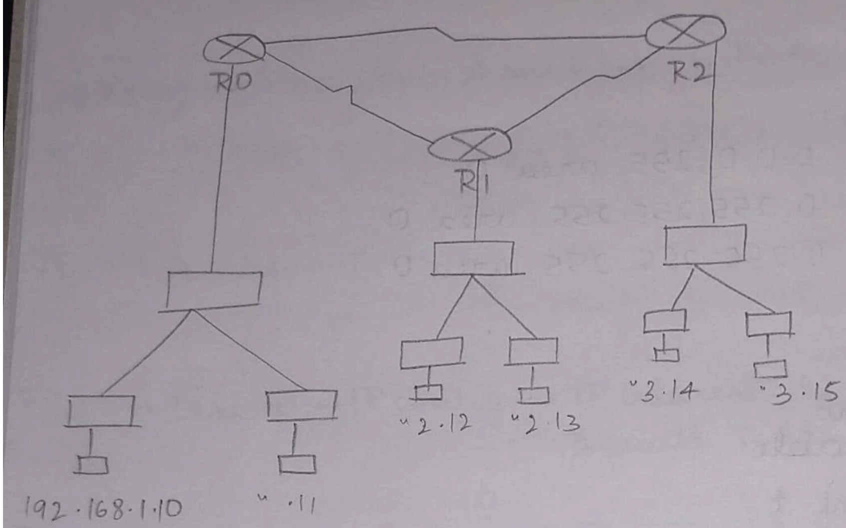
Router 1 → CLI → no enable

```
conf t
hostname R1
int Fa0/0
ip address 192 168 2 1 255 255 255 0
no shutdown
int Se0/1/0
ip add 10 0 0 2 255 0 0 0
no shutdown
int Se0/1/1
ip add 11 0 0 1 255 0 0 0
clock rate 64000
no shutdown
exit
router ospf 1
network 192 168 2 0 0 0 0 255 area 0
network 10 0 0 0 0 255 255 255 area 0
network 11 0 0 0 0 255 255 255 area 0
exit
exit
```

Repeat the same for Router 2.

Packet sent from R0 to R1
R1 to R2
R0 to R2

Observation -
OSPF dynamically established neighbor relations btwn routers within same area.



06/11

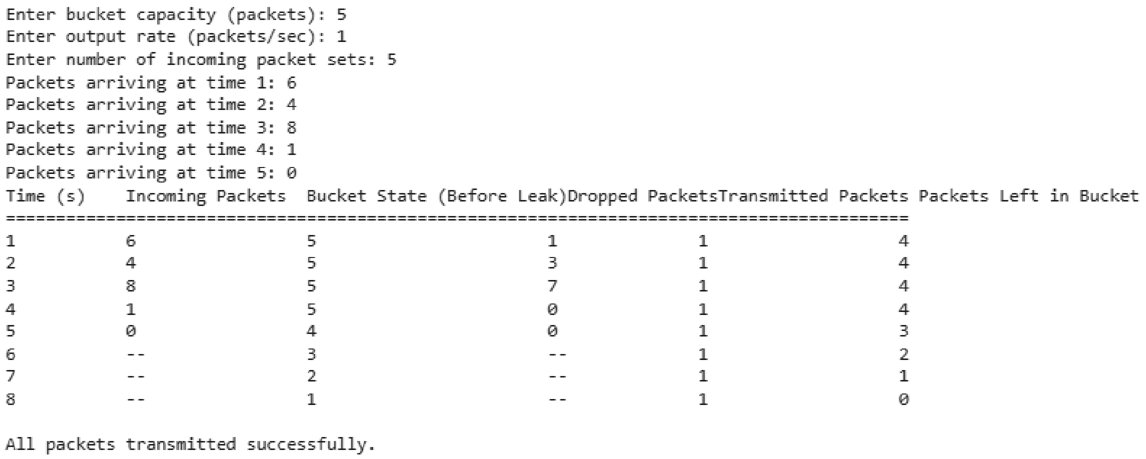
Program – 12:

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

Program:

```
def leaky_bucket(bucket_capacity, output_rate, incoming_packets):
    stored = 0  # current number of packets in the bucket

    # Print Table Header
    print(f"{'Time (s)':<12}{'Incoming Packets':<18}{'Bucket State (Before Leak)':<24}{'Dropped Packets':<15}{'Transmitted Packets':<20}{'Packets Left in Bucket'}")
    print("="*90)

    for time, packets in enumerate(incoming_packets, start=1):
        # Handle overflow: if incoming packets cause bucket overflow
        if packets + stored > bucket_capacity:
            dropped = (packets + stored) - bucket_capacity
            stored = bucket_capacity
        else:
            dropped = 0
            stored += packets

        # Transmit packets at output rate
        transmitted = min(stored, output_rate)
        stored -= transmitted

        # Print row for the table
        print(f"{time:<12}{packets:<18}{stored + transmitted:<24}{dropped:<15}{transmitted:<20}{stored}")

    # Empty remaining packets in the bucket after incoming packets are done
    while stored > 0:
        time += 1
        transmitted = min(stored, output_rate)
        stored -= transmitted

        # Print row for remaining packets
        print(f"{time:<12}{'--':<18}{stored + transmitted:<24}{'--':<15}{transmitted:<20}{stored}")

    print("\nAll packets transmitted successfully.")

# ---- Main Program ----
if __name__ == "__main__":
    bucket_capacity = int(input("Enter bucket capacity (packets): "))
    output_rate = int(input("Enter output rate (packets/sec): "))

    n = int(input("Enter number of incoming packet sets: "))
    incoming_packets = []

    for i in range(n):
        packets = int(input(f"Packets arriving at time {i + 1}: "))
        incoming_packets.append(packets)

    leaky_bucket(bucket_capacity, output_rate, incoming_packets)
```

Output:



```
Enter bucket capacity (packets): 5
Enter output rate (packets/sec): 1
Enter number of incoming packet sets: 5
Packets arriving at time 1: 6
Packets arriving at time 2: 4
Packets arriving at time 3: 8
Packets arriving at time 4: 1
Packets arriving at time 5: 0
Time (s)    Incoming Packets  Bucket State (Before Leak)Dropped PacketsTransmitted Packets Packets Left in Bucket
==========================================================================================
1           6                 5                       1              1                   4
2           4                 5                       3              1                   4
3           8                 5                       7              1                   4
4           1                 5                       0              1                   4
5           0                 4                       0              1                   3
6           --                3                       --             1                   2
7           --                2                       --             1                   1
8           --                1                       --             1                   0

All packets transmitted successfully.
```

Program – 13:

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

Program:

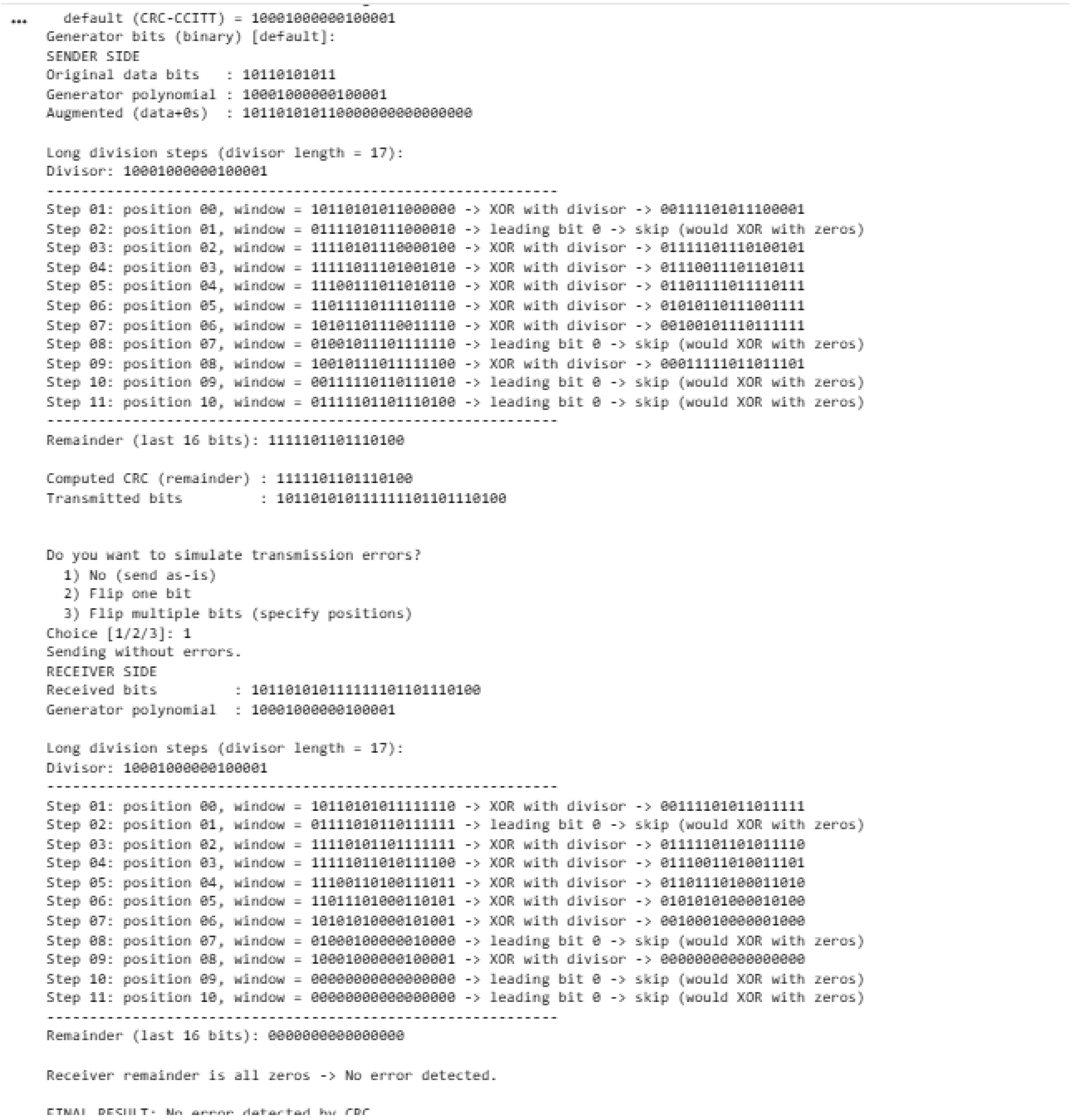
```
#!/usr/bin/env python3
"""
CRC-CCITT (16-bit) interactive demonstration.
Shows step-by-step binary long division at sender and receiver.
"""

def validate_binary_str(s, name="input"):
    if not s:
        raise ValueError(f"{name} cannot be empty.")
    if any(c not in '01' for c in s):
        raise ValueError(f"{name} must contain only '0' and '1' characters.")
    return s

def xor(a, b):
    """XOR two binary strings of same length, return result string"""
    return ".join('0' if x == y else '1' for x, y in zip(a, b))

def long_division_show(dividend, divisor, show_steps=True):
    """
    Perform binary long division (modulo-2) and return remainder.
    If show_steps True, prints the division steps.
    dividend/divisor are strings of '0'/'1'. divisor's leading bit must be '1'.
    """
    n = len(divisor)
    # Work on a mutable list for the dividend bits
    work = list(dividend)
    if show_steps:
        print("\nLong division steps (divisor length = {}):".format(n))
        print("Divisor: {}".format(divisor))
        print("-" * 60)

    for i in range(len(dividend) - n + 1):
        # show current window
        window = ".join(work[i:i+n])
        if show_steps:
            print(f"Step {i+1:02d}: position {i:02d}, window = {window}", end=")
        if window[0] == '1':
            # XOR with divisor
            new_bits = xor(window, divisor)
            if show_steps:
                print(f" -> XOR with divisor -> {new_bits}")
            # write back
            work[i:i+n] = list(new_bits)
        else:
            if show_steps:
                print(" -> leading bit 0 -> skip (would XOR with zeros)")
            # effectively XOR with zeros; we can skip
            # (no change to work)
    remainder = ".join(work[-(n-1):]) if n > 1 else '0'
    if show_steps:
        print("-" * 60)
```

```
        print(f"Remainder (last {n-1} bits): {remainder}\n")
    return remainder

def compute_crc_sender(data_bits, generator_bits, show_steps=True):
    """
    Compute CRC for data_bits using generator_bits.
    Returns the CRC (remainder) and the transmitted bits (data + crc).
    """
    validate_binary_str(data_bits, "Data bits")
    validate_binary_str(generator_bits, "Generator bits")
    if generator_bits[0] != '1':
        raise ValueError("Generator polynomial must start with '1' (highest-order bit).")

    k = len(generator_bits) - 1  # CRC length in bits
    # Append k zeros to data
    augmented = data_bits + '0' * k
    if show_steps:
        print("SENDER SIDE")
        print("Original data bits   : {}".format(data_bits))
        print("Generator polynomial : {}".format(generator_bits))
        print("Augmented (data+0s)  : {}".format(augmented))
    remainder = long_division_show(augmented, generator_bits, show_steps=show_steps)
    # CRC is remainder; transmitted frame is original data + remainder
    transmitted = data_bits + remainder
    if show_steps:
        print(f"Computed CRC (remainder) : {remainder}")
        print(f"Transmitted bits         : {transmitted}\n")
    return remainder, transmitted

def check_crc_receiver(received_bits, generator_bits, show_steps=True):
    """
    Receiver side: check received_bits using generator_bits.
    Returns True if no error detected (remainder all zeros), False otherwise.
    """
    validate_binary_str(received_bits, "Received bits")
    validate_binary_str(generator_bits, "Generator bits")
    if generator_bits[0] != '1':
        raise ValueError("Generator polynomial must start with '1' (highest-order bit).")
    if show_steps:
        print("RECEIVER SIDE")
        print("Received bits         : {}".format(received_bits))
        print("Generator polynomial  : {}".format(generator_bits))
    remainder = long_division_show(received_bits, generator_bits, show_steps=show_steps)
    ok = all(ch == '0' for ch in remainder)
    if show_steps:
        if ok:
            print("Receiver remainder is all zeros -> No error detected.")
        else:
            print("Receiver remainder is NOT all zeros -> Error detected!")
    return ok, remainder

def demo_interactive():
    print("=== CRC-CCITT (16-bit) Demo ===")
    print("Enter data bits (binary), e.g. 11010011101100")
    data_bits = input("Data bits: ").strip()
```

```
# Default CRC-CCITT generator: x^16 + x^12 + x^5 + 1
default_generator = "10001000000100001"  # 17 bits (degree 16 down to 0)
print("\nPress Enter to use default CRC-CCITT generator:")
print(f"  default (CRC-CCITT) = {default_generator}")
gen_in = input("Generator bits (binary) [default]: ").strip()
generator_bits = gen_in if gen_in else default_generator

# Sender computes CRC and shows steps
try:
    crc, transmitted = compute_crc_sender(data_bits, generator_bits, show_steps=True)
except ValueError as e:
    print("Input error:", e)
    return

# Ask if we should flip bits (simulate errors)
print("\nDo you want to simulate transmission errors?")
print("  1) No (send as-is)")
print("  2) Flip one bit")
print("  3) Flip multiple bits (specify positions)")
choice = input("Choice [1/2/3]: ").strip() or '1'

received = transmitted
if choice == '2':
    pos = input(f"Enter bit position to flip (0..{len(transmitted)-1}), 0 is leftmost: ").strip()
    try:
        p = int(pos)
        if not (0 <= p < len(transmitted)):
            raise ValueError
        b = '1' if transmitted[p] == '0' else '0'
        received = transmitted[:p] + b + transmitted[p+1:]
        print(f"Flipped bit at position {p}. Received bits: {received}")
    except:
        print("Invalid position; no flip performed.")
elif choice == '3':
    pos_list = input(f"Enter space-separated positions to flip (0..{len(transmitted)-1}): ").strip().split()
    changed = list(transmitted)
    for pos in pos_list:
        try:
            p = int(pos)
            if 0 <= p < len(transmitted):
                changed[p] = '1' if changed[p] == '0' else '0'
        except:
            pass
    received = ''.join(changed)
    print(f"Received bits after flips: {received}")
else:
    print("Sending without errors.")

# Receiver checks CRC
ok, rem = check_crc_receiver(received, generator_bits, show_steps=True)
if ok:
    print("\nFINAL RESULT: No error detected by CRC.")
else:
    print("\nFINAL RESULT: Error detected by CRC.")
```

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

Output:



```
...   default (CRC-CCITT) = 10001000000100001
    Generator bits (binary) [default]:
    SENDER SIDE
    Original data bits   : 10110101011
    Generator polynomial : 10001000000100001
    Augmented (data+0s)  : 101101010110000000000000000

    Long division steps (divisor length = 17):
    Divisor: 10001000000100001
    ------------------------------------------------------------
    Step 01: position 00, window = 10110101011000000 -> XOR with divisor -> 00111101011100001
    Step 02: position 01, window = 01111010111000010 -> leading bit 0 -> skip (would XOR with zeros)
    Step 03: position 02, window = 11110101110000100 -> XOR with divisor -> 01111101110100101
    Step 04: position 03, window = 11111011101001010 -> XOR with divisor -> 01110011101101011
    Step 05: position 04, window = 11100111011010110 -> XOR with divisor -> 01101111011110111
    Step 06: position 05, window = 11011110111101110 -> XOR with divisor -> 01010110111001111
    Step 07: position 06, window = 10101101110011110 -> XOR with divisor -> 00100101110111111
    Step 08: position 07, window = 01001011101111110 -> leading bit 0 -> skip (would XOR with zeros)
    Step 09: position 08, window = 10010111011111100 -> XOR with divisor -> 00011111011011101
    Step 10: position 09, window = 00111110110111010 -> leading bit 0 -> skip (would XOR with zeros)
    Step 11: position 10, window = 01111101101110100 -> leading bit 0 -> skip (would XOR with zeros)
    ------------------------------------------------------------
    Remainder (last 16 bits): 1111101101110100

    Computed CRC (remainder) : 1111101101110100
    Transmitted bits         : 101101010111111101101110100


    Do you want to simulate transmission errors?
      1) No (send as-is)
      2) Flip one bit
      3) Flip multiple bits (specify positions)
    Choice [1/2/3]: 1
    Sending without errors.
    RECEIVER SIDE
    Received bits         : 101101010111111101101110100
    Generator polynomial  : 10001000000100001

    Long division steps (divisor length = 17):
    Divisor: 10001000000100001
    ------------------------------------------------------------
    Step 01: position 00, window = 10110101011111110 -> XOR with divisor -> 00111101011011111
    Step 02: position 01, window = 01111010110111111 -> leading bit 0 -> skip (would XOR with zeros)
    Step 03: position 02, window = 11110101101111111 -> XOR with divisor -> 01111101101011110
    Step 04: position 03, window = 11111011010111100 -> XOR with divisor -> 01110011010011101
    Step 05: position 04, window = 11100110100111011 -> XOR with divisor -> 01101110100011010
    Step 06: position 05, window = 11011101000110101 -> XOR with divisor -> 01010101000010100
    Step 07: position 06, window = 10101010000101001 -> XOR with divisor -> 00100010000001000
    Step 08: position 07, window = 01000100000010000 -> leading bit 0 -> skip (would XOR with zeros)
    Step 09: position 08, window = 10001000000100001 -> XOR with divisor -> 00000000000000000
    Step 10: position 09, window = 00000000000000000 -> leading bit 0 -> skip (would XOR with zeros)
    Step 11: position 10, window = 00000000000000000 -> leading bit 0 -> skip (would XOR with zeros)
    ------------------------------------------------------------
    Remainder (last 16 bits): 0000000000000000

    Receiver remainder is all zeros -> No error detected.

    FINAL RESULT: No error detected by CRC
```

Program – 14:

Aim: 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.

Program:

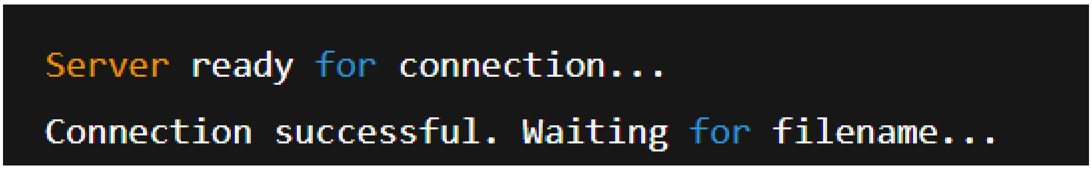
CLIENTSIDE:

```
 package pgm7;
import java.net.*;
import java.io.*;

public class TCPC {
   public static void main(String[] args) throws Exception {
      Socket sock = new Socket("127.0.0.1", 4000);

      System.out.println("Enter the filename:");
      BufferedReader keyRead = new BufferedReader(new InputStreamReader(System.in));

      String fname = keyRead.readLine();

      OutputStream ostream = sock.getOutputStream();
      PrintWriter pwrite = new PrintWriter(ostream, true);

      // Send file name to server
      pwrite.println(fname);

      InputStream istream = sock.getInputStream();
      BufferedReader socketRead = new BufferedReader(new InputStreamReader(istream));

      String str;
      while ((str = socketRead.readLine()) != null) {
         System.out.println(str);
      }

      pwrite.close();
      socketRead.close();
      keyRead.close();
      sock.close();
   }
}


import java.net.*;
import java.io.*;

public class TCPS {
   public static void main(String[] args) throws Exception {

      ServerSocket sersock = new ServerSocket(4000);
      System.out.println("Server ready for connection...");

      Socket sock = sersock.accept();
      System.out.println("Connection successful. Waiting for filename...");

      InputStream istream = sock.getInputStream();
      BufferedReader fileRead = new BufferedReader(new InputStreamReader(istream));
```

```
        String fname = fileRead.readLine();     // Read filename from client
        BufferedReader contentRead = new BufferedReader(new FileReader(fname));

        OutputStream ostream = sock.getOutputStream();
        PrintWriter pwrite = new PrintWriter(ostream, true);

        String str;
        while ((str = contentRead.readLine()) != null) {
            pwrite.println(str);
        }

        sock.close();
        sersock.close();
        pwrite.close();
        fileRead.close();
        contentRead.close();
    }
}
```

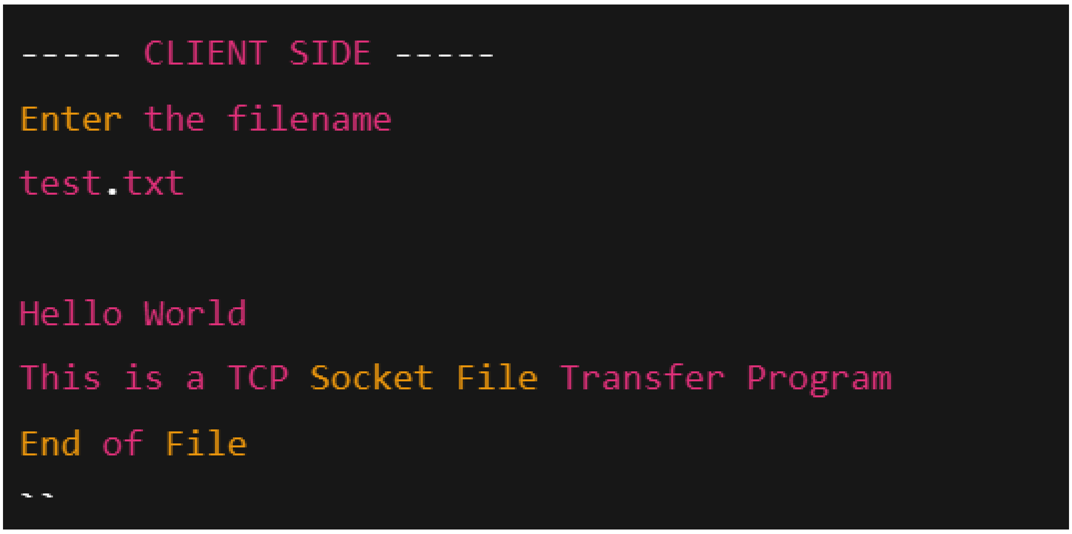
Output:

SERVER OUTPUT:



```
Server ready for connection...
Connection successful. Waiting for filename...
```

CLIENT OUTPUT:



```
----- CLIENT SIDE -----
Enter the filename
test.txt


Hello World
This is a TCP Socket File Transfer Program
End of File
``
```

Program – 15:

Aim: 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.

Program:

```
package pgm8;

import java.io.*;
import java.net.*;

class UDPClient {
   public static void main(String[] args) throws Exception {

      BufferedReader inFromUser =
         new BufferedReader(new InputStreamReader(System.in));

      DatagramSocket clientSocket = new DatagramSocket();
      InetAddress IPAddress = InetAddress.getByName("localhost");

      byte[] sendData = new byte[1024];
      byte[] receiveData = new byte[1024];

      System.out.println("Enter the string to be converted to Uppercase:");
      String sentence = inFromUser.readLine();

      sendData = sentence.getBytes();

      DatagramPacket sendPacket =
         new DatagramPacket(sendData, sendData.length, IPAddress, 9876);

      clientSocket.send(sendPacket);

      DatagramPacket receivePacket =
         new DatagramPacket(receiveData, receiveData.length);

      clientSocket.receive(receivePacket);

      String modifiedSentence = new String(receivePacket.getData());

      System.out.println("FROM SERVER: " + modifiedSentence);

      clientSocket.close();
   }
}

package pgm8;

import java.net.*;

class UDPServer {
   public static void main(String[] args) throws Exception {

      DatagramSocket serverSocket = new DatagramSocket(9876);

      byte[] receiveData = new byte[1024];
```

```
    byte[] sendData = new byte[1024];

    while (true) {
      System.out.println("Server is Up");

      DatagramPacket receivePacket =
        new DatagramPacket(receiveData, receiveData.length);

      serverSocket.receive(receivePacket);

      String sentence = new String(receivePacket.getData());
      System.out.println("RECEIVED: " + sentence.trim());

      InetAddress IPAddress = receivePacket.getAddress();
      int port = receivePacket.getPort();

      String capitalizedSentence = sentence.toUpperCase();

      sendData = capitalizedSentence.getBytes();

      DatagramPacket sendPacket =
        new DatagramPacket(sendData, sendData.length, IPAddress, port);

      serverSocket.send(sendPacket);
    }
  }
}
```

Output:

```
----- SERVER SIDE -----
Server is Up
RECEIVED: hello world
Server is Up

----- CLIENT SIDE -----
Enter the string to be converted to Uppercase:
hello world
FROM SERVER: HELLO WORLD
```