VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB RECORD

Computer Network Lab (23CS5PCCON)

Submitted by

AARUSH GARG (1BM23CS004)

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 **Aarush Garg (1BM23CS004),** who is 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.Praveen N. Assistant Professor Department of CSE, BMSCE	Dr. Kavitha Sooda Professor & HOD Department of CSE, BMSCE

Index

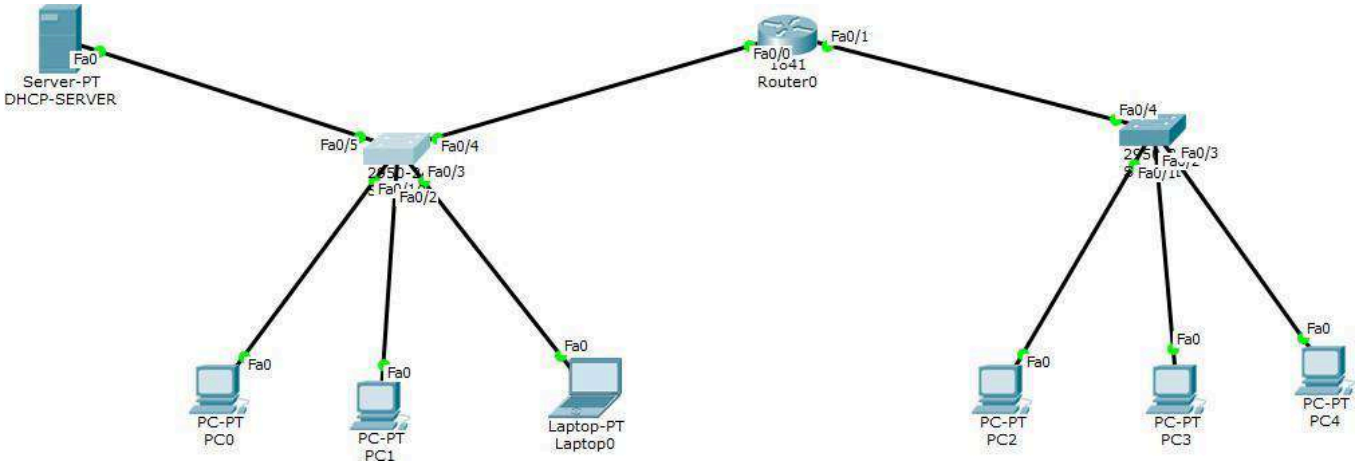
Sl. No.	Experiment Title	Page No.
1	Configure DHCP within a LAN and outside LAN.	4
2	Configure Web Server, DNS within a LAN.	7
3	Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply	10
4	Configure default route, static route to the Router	12
5	To understand the operation of TELNET by accessing the router in server room from a PC in IT office.	17
6	Configure RIP routing Protocol in Routers	21
7	To construct a WLAN and make the nodes communicate wirelessly	23
8	To construct simple LAN and understand the concept and operation of Address Resolution Protocol (ARP)	25
9	To construct a VLAN and make the PC's communicate among a VLAN	27
10	Configure OSPF routing protocol	29
11	Create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices and demonstrate ping message.	33
12	Demonstrate the TTL/ Life of a Packet	34
13	Write a program for error detecting code using CRC-CCITT (16-bits).	35
14	Write a program for congestion control using Leaky bucket algorithm.	38
15	Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.	42
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.	44

Github Link:
https://github.com/Aarush004/Computer-Network

Program – 1:

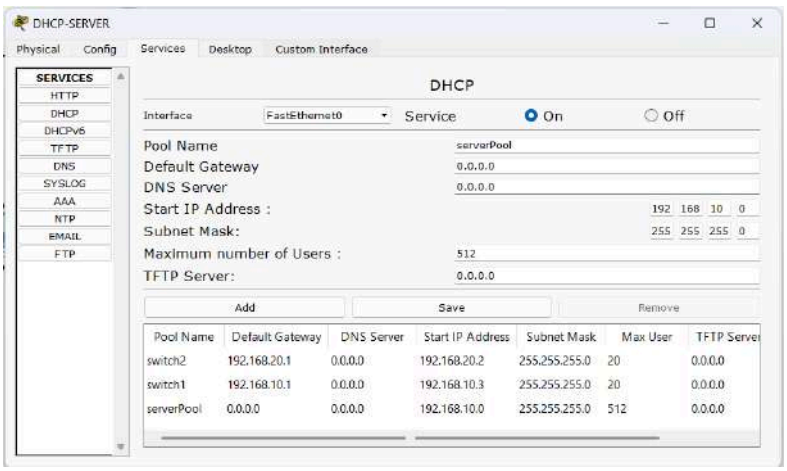
Aim: Configure DHCP within a LAN and outside LAN.

Topology:



Procedure:

1. Configure DHCP Server:
   in DHCP server go to Desktop>IP-Config, assign static IP – 192.168.10.2 and gateway 192.168.10.1
2. Open Services>DHCP and add following two DHCP pool:
   (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: 192.168.10.2
   Subnet Mask: 255.255.255.0
   Max Users: 20



3. Configure Router
   i. Router>enable
   ii. Router#conf t

(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. do write memory
   viii. Router(config-if)# exit

(Outside Lan)

   ix. Router(config)# int Fa0/1
   x. Router(config-if)# ip address 192.168.20.1 255.255.255.0
   xi. Router(config-if)# ip helper-address 192.168.10.2
   xii. Router(config-if)# no shutdown
   xiii. do write memory
   xiv. Router(config-if)# exit

   xv. Router(config)# exit
   xvi. Router# write memory

Output

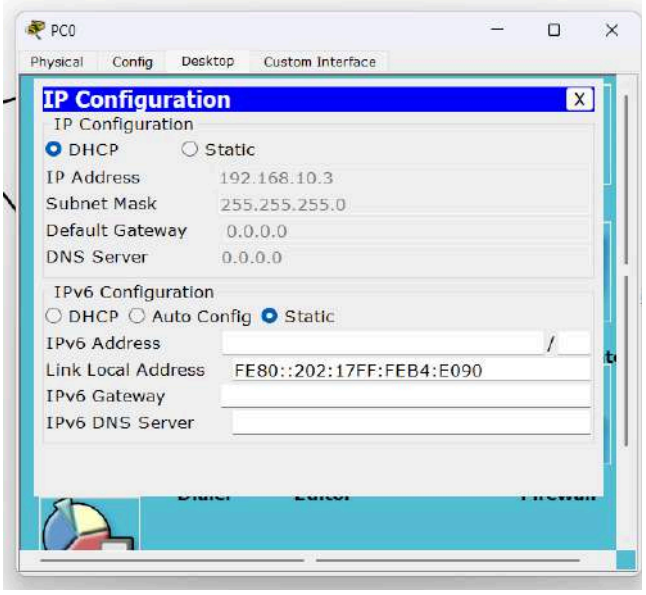


Fig 1. Ip address assigned by DHCP server within Lan (PC0)

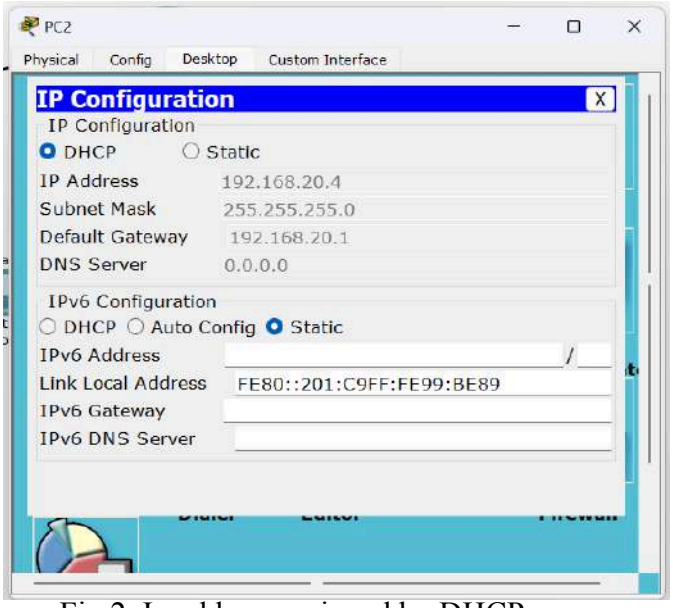
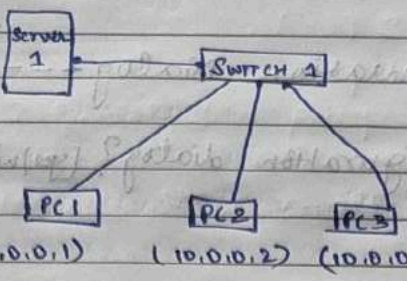
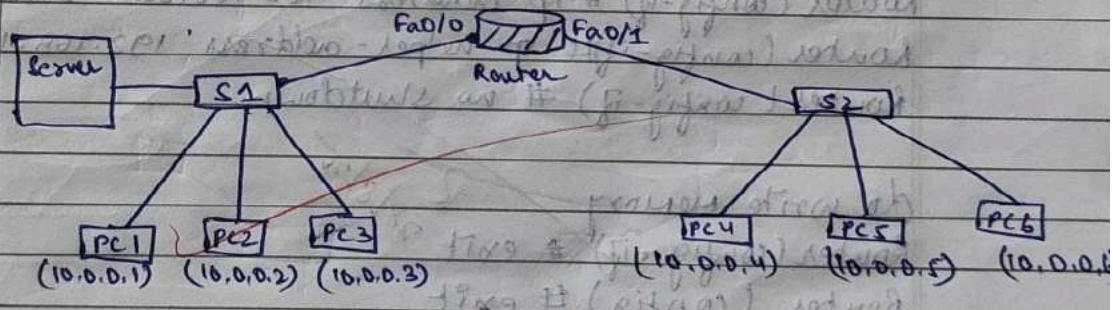


Fig 2. Ip address assigned by DHCP server outside Lan (PC2)

Observation:



- This is a client-server LAN using a switch
- There all 3 devices connected to a switch.
- A server is connected to switch.
- All the connections are been made through ethernet cables
- The server is providing centralized services like DHCP to all the PCs

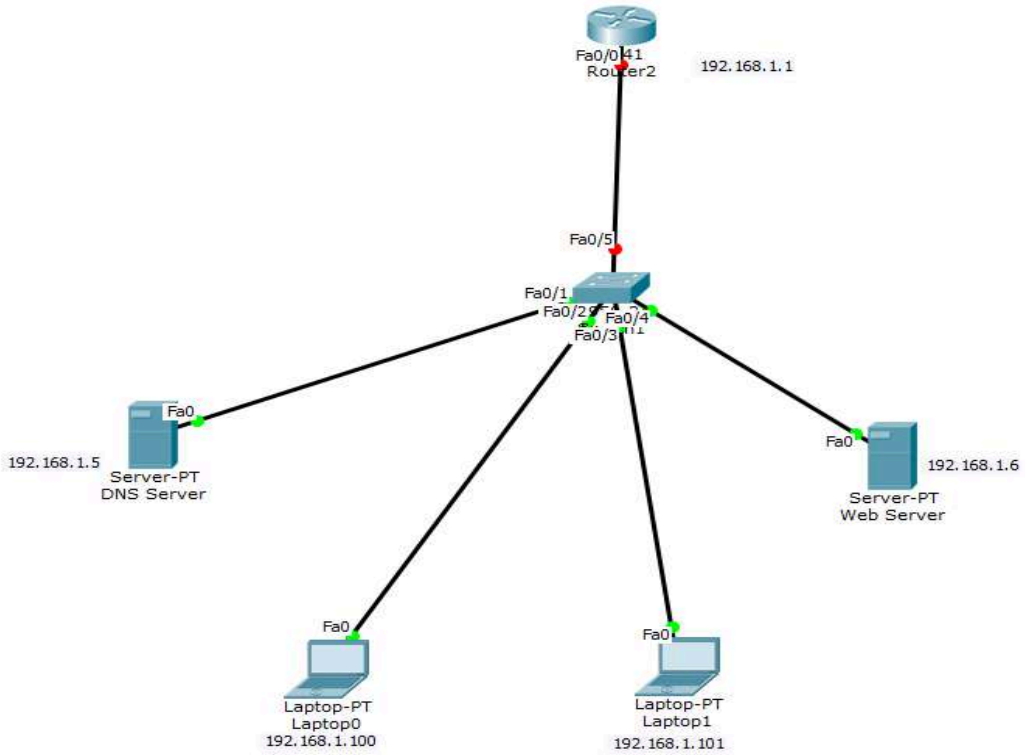


- This diagram represents a LAN-WAN interconnected topology consisting of 2 LANs connected through a router.
- The router enables communication b/w 2 LANs
- The server in LAN 1 can provide services to all devices in both networks.

Program – 2:

Aim: Configure Web Server, DNS within a LAN.

Topology:

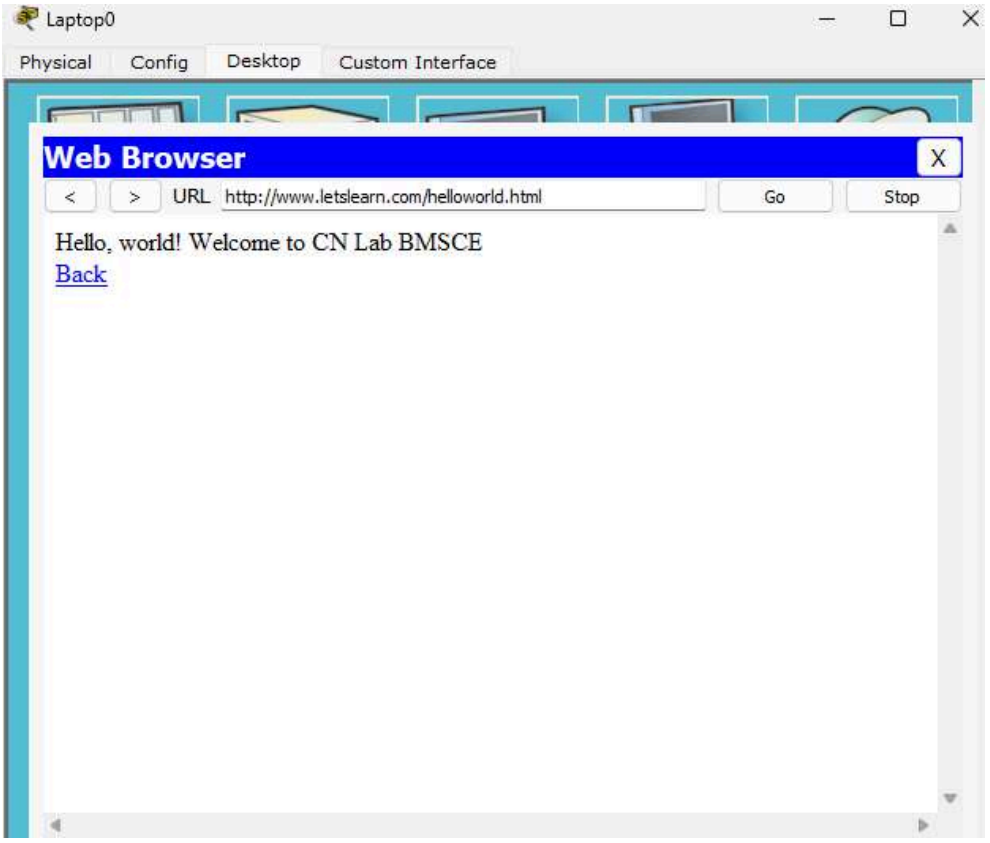


Procedure:

1. Configure DNS Server:
   IPv4 Address : 192.168.1.5
   Subnet Mask : 255.255.255.0
   Default Gateway : 192.168.1.1
   DNS Server : 192.168.1.5
2. Configure Web Server:
   IPv4 Address : 192.168.1.6
   Subnet Mask : 255.255.255.0
   Default Gateway : 192.168.1.1
   DNS Server : 192.168.1.5
3. Configure PC0 Server:
   IPv4 Address : 192.168.1.100
   Subnet Mask : 255.255.255.0
   Default Gateway : 192.168.1.1
   DNS Server : 192.168.1.5
4. Configure PC1 Server:
   IPv4 Address : 192.168.1.5
   Subnet Mask : 255.255.255.0
   Default Gateway : 192.168.1.1
   DNS Server : 192.168.1.5

5. In Web Server, activate HTTP and HTTPS services to ON. Edit any 1 file manager (helloworld.html).
6. Then in DNS Server, turn ON the DNS services. Add name www.letslearn.com with Type : A Record. In the Address field add web server's address (192.168.1.6). Click Add->Save.
7. Now go to PC0's command prompt and give command :
   PING 192.168.1.5
   PING 192.168.1.6
   (to check connection is done successfully or not)
8. Then go to Web Browser in PC0 and type URL : http://www.letslearn.com
9. Next click on 'A Small page' link.

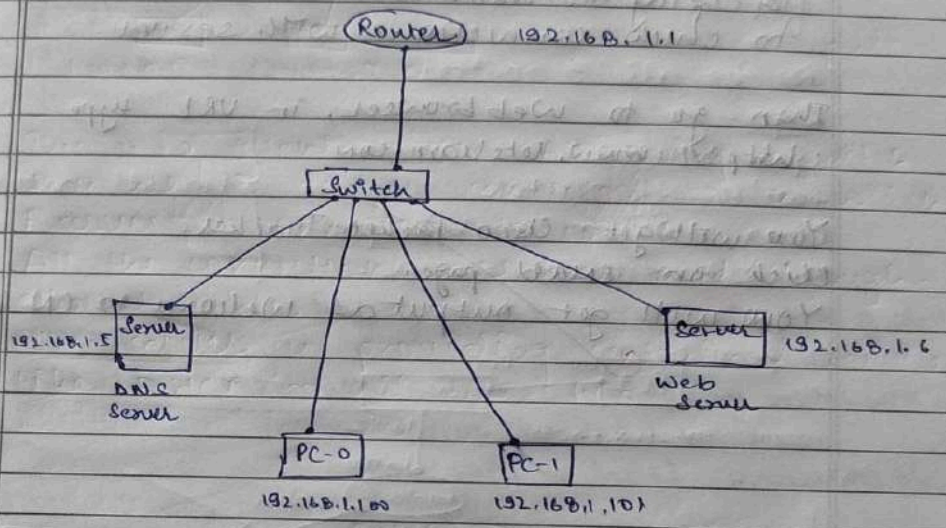
Output



Observation:

Configure Web server & DNS within a LAN

Default Gateway → Router IP Address



In PC-0 and PC-1 add IP Addresses.
Give DNS server's address as 192.168.1.5.
In default Gateway, give router's address.

In Web Server, turn on HTTP & HTTPS in services.
Then edit any 1 file manager.

Now in DNS server do IP configuration.
Turn on DNS Services.
Add name "letsLearn.com" with Web server's address.

Now Go to PC-0 & PC-1.
In command prompt.
give command ping 192.168.1.5
& ping 192.168.1.6
to check connections with server.

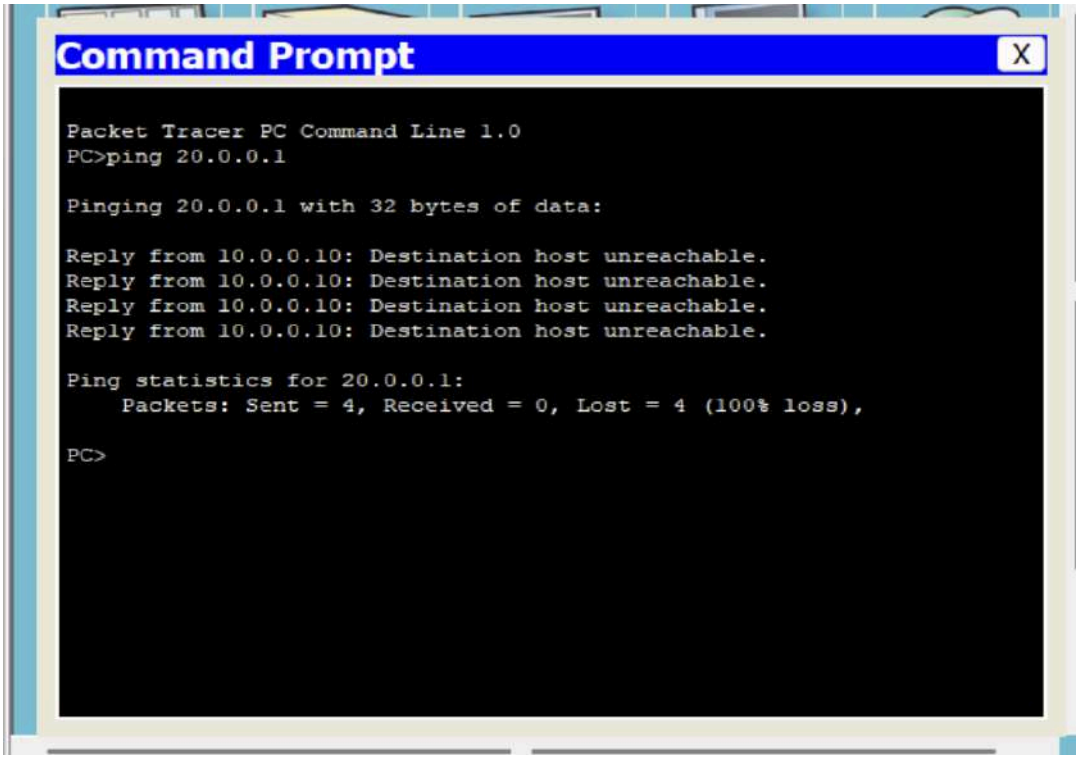
Then go to web browser, in URL type
http://www.letslearn.com

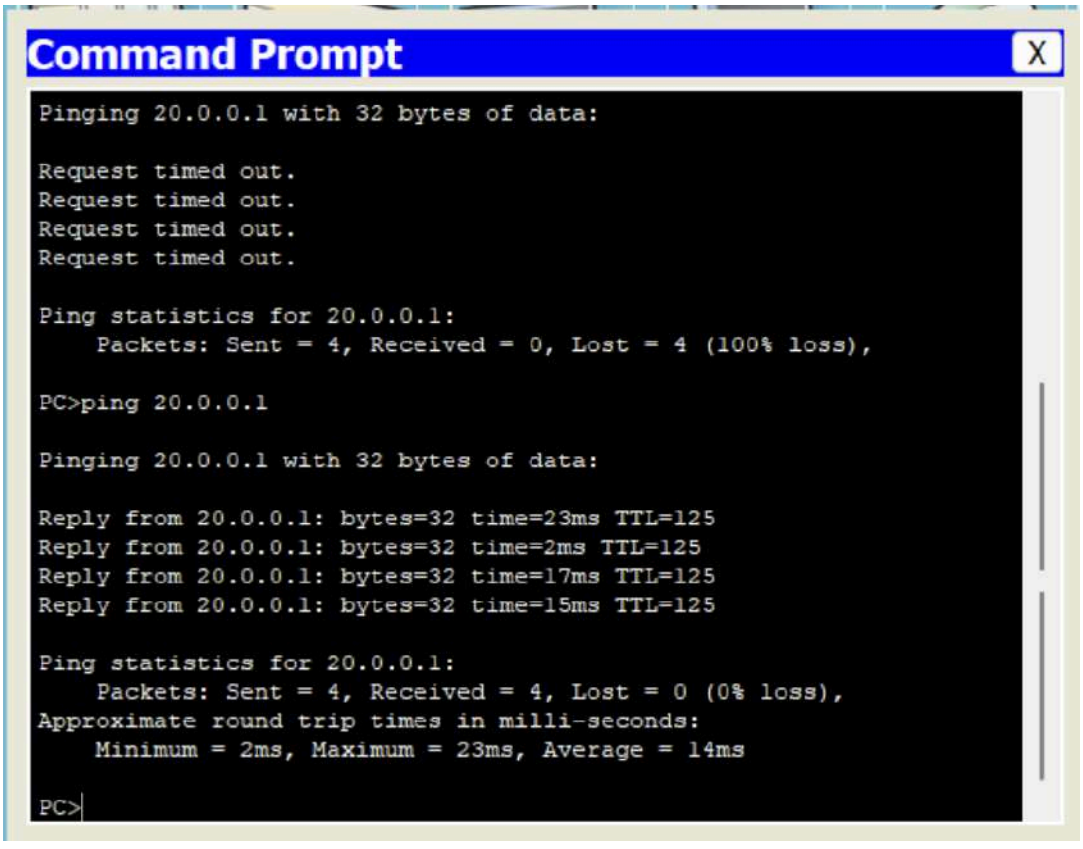
You will get Cisco Packet Tracker.
Click on small page.
You will get output as Welcome to CN lab, RMIT

Program – 3:

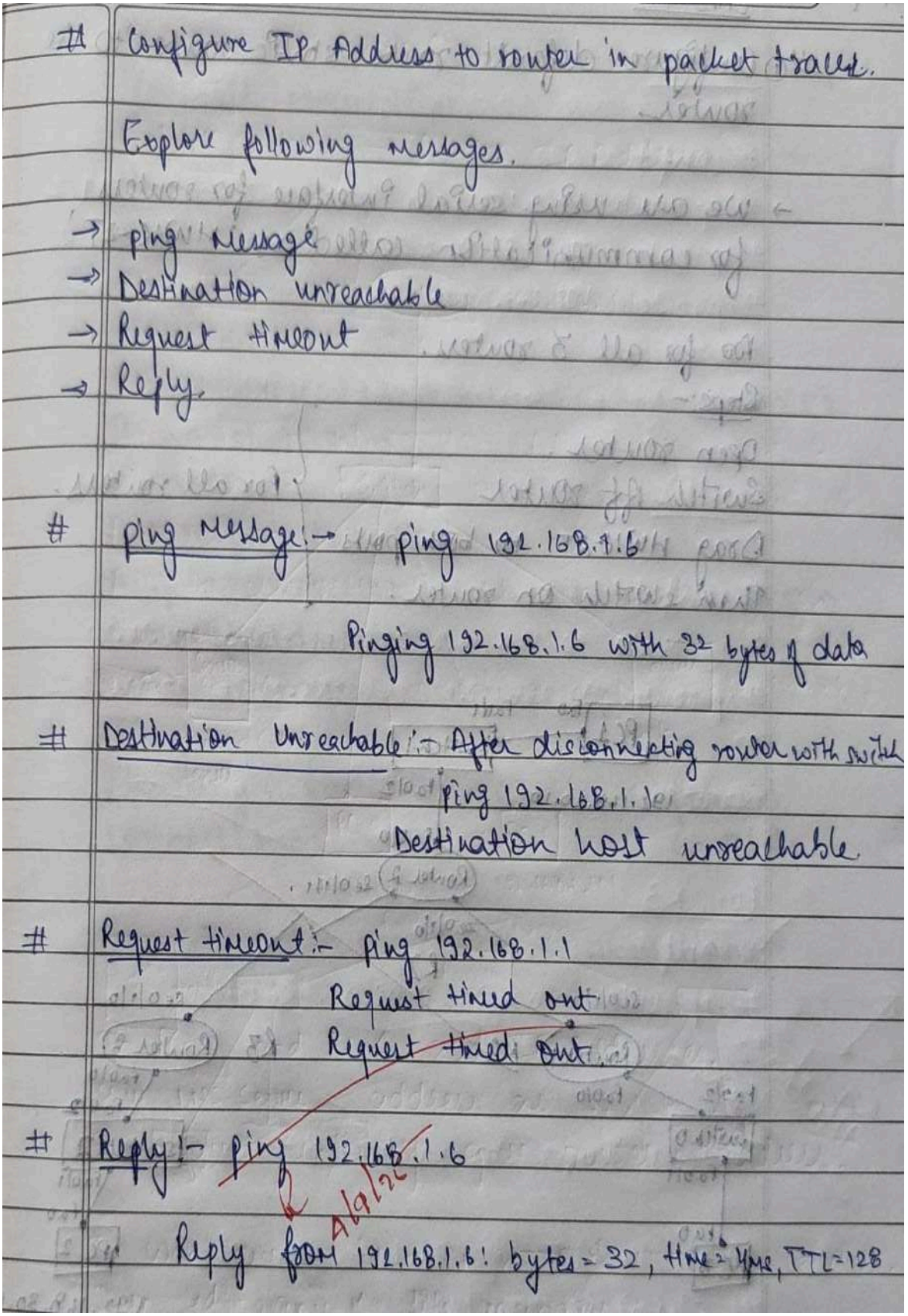
Aim: Configure IP address to routers in packet tracer. Explore the following messages: ping responses, destination unreachable, request timed out, reply

Output





Observation:



Configure IP Address to router in packet tracer.

Explore following messages.

→ ping message
→ Destination unreachable
→ Request timeout
→ Reply.

ping message:→ ping 192.168.1.6

Pinging 192.168.1.6 with 32 bytes of data

Destination Unreachable:- After disconnecting router with switch

ping 192.168.1.1

Destination host unreachable.

Request timeout:- ping 192.168.1.1

Request timed out

Request timed out.

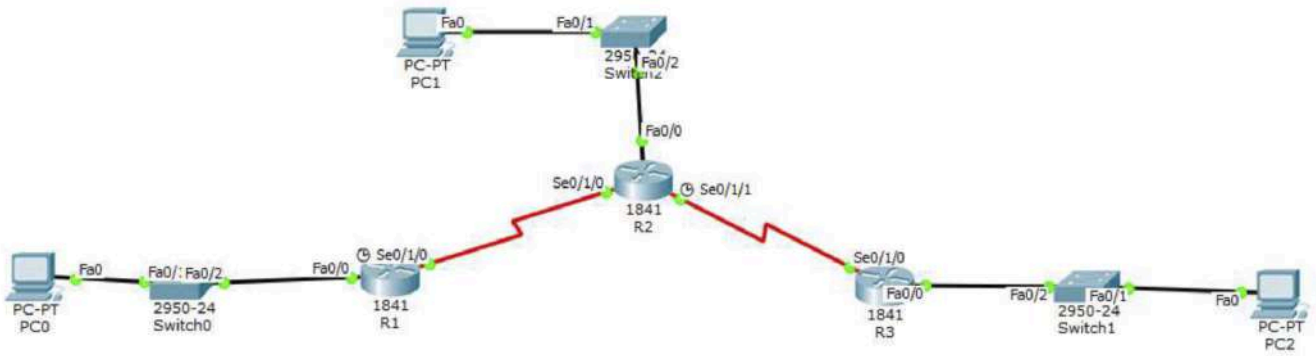
Reply:- ping 192.168.1.6

Reply from 192.168.1.6: bytes=32, time=4ms, TTL=128

Program – 4:

Aim: Configure default route, static route to the Router

Topology:



Procedure:

1. Open R1:
    a. Switch off R1
    b. Drag 'HWIC-2T' module to both ports.
    c. Switch on R1
2. Repeat step 1 for R2, R3
3. Configure R1:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# int Se0/1/0
    d. Router(config-if)# ip address 172.16.1.1 255.255.255.252
    e. Router(config-if)# no shutdown
    f. Router(config-if)# exit
    g. Router(config)# int Fa0/0
    h. Router(config-if)# ip address 192.168.10.0 255.255.255.0
    i. Router(config-if)# no shutdown
    j. Write memory
    k. Router(config-if)# exit
    l. Router(config)# exit
    m. Write memory
4. Configure R2:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# int Se0/1/0
    d. Router(config-if)# ip address 172.16.1.2 255.255.255.252
    e. Router(config-if)# no shutdown
    f. Router(config-if)# exit
    g. Router(config)# int Fa0/0
    h. Router(config-if)# ip address 192.168.20.0 255.255.255.0
    i. Router(config-if)# no shutdown
    j. Router(config-if)# exit
    k. Router(config)# int Se0/1/1
    l. Router(config-if)# ip address 192.168.2.1 255.255.255.252
    m. Router(config-if)# no shutdown
    n. Write memory
    o. Router(config-if)# exit

p. Router(config)# exit
q. Write memory

5. Configure R3:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# int Se0/1/0
    d. Router(config-if)# ip address 172.16.2.2 255.255.255.252
    e. Router(config-if)# no shutdown
    f. Router(config-if)# exit
    g. Router(config)# int Fa0/0
    h. Router(config-if)# ip address 192.168.30.0 255.255.255.0
    i. Router(config-if)# no shutdown
    j. Write memory
    k. Router(config-if)# exit
    l. Router(config)# exit
    m. Write memory
6. Next configure IP address of all 3 PCs:{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}
7. Next in command line of each router establish path:
8. Configure R1:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# ip route 192.168.20.0 255.255.255.0 172.16.1.2
    d. Router(config)# ip route 172.16.2.0 255.255.255.252 172.168.1.2
    e. Router(config)# ip route 192.168.30.0 255.255.255.0 172.168.1.2
    n. Router(config)# <ctrl + z>
    o. wr
9. Configure R2:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# ip route 192.168.10.0 255.255.255.0 172.16.1.1
    d. Router(config)# ip route 192.168.30.0 255.255.255.0 172.168.2.2
    e. Router(config)# <ctrl + z>
    f. wr
10. Configure R3:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# ip route 0.0.0.0 0.0.0.0 Se0/1/0 //(default)
    d. Router(config)# <ctrl + z>
    e. wr
11. Next in each CLI of router do show ip route {to check successful connection}
12. Next in each PCs command prompt do {All Success}
    a. Ping 192.168.10.1
    b. Ping 192.168.20.1
    c. Ping 192.168.30.1

Output

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC0	PC1	ICMP		0.000	N	0	(edit)	(delete)
	Successful	PC0	PC1	ICMP		0.000	N	1	(edit)	(delete)
	Successful	PC2	PC1	ICMP		0.000	N	2	(edit)	(delete)

Observation

#1 Configure default route, static route to the routers.

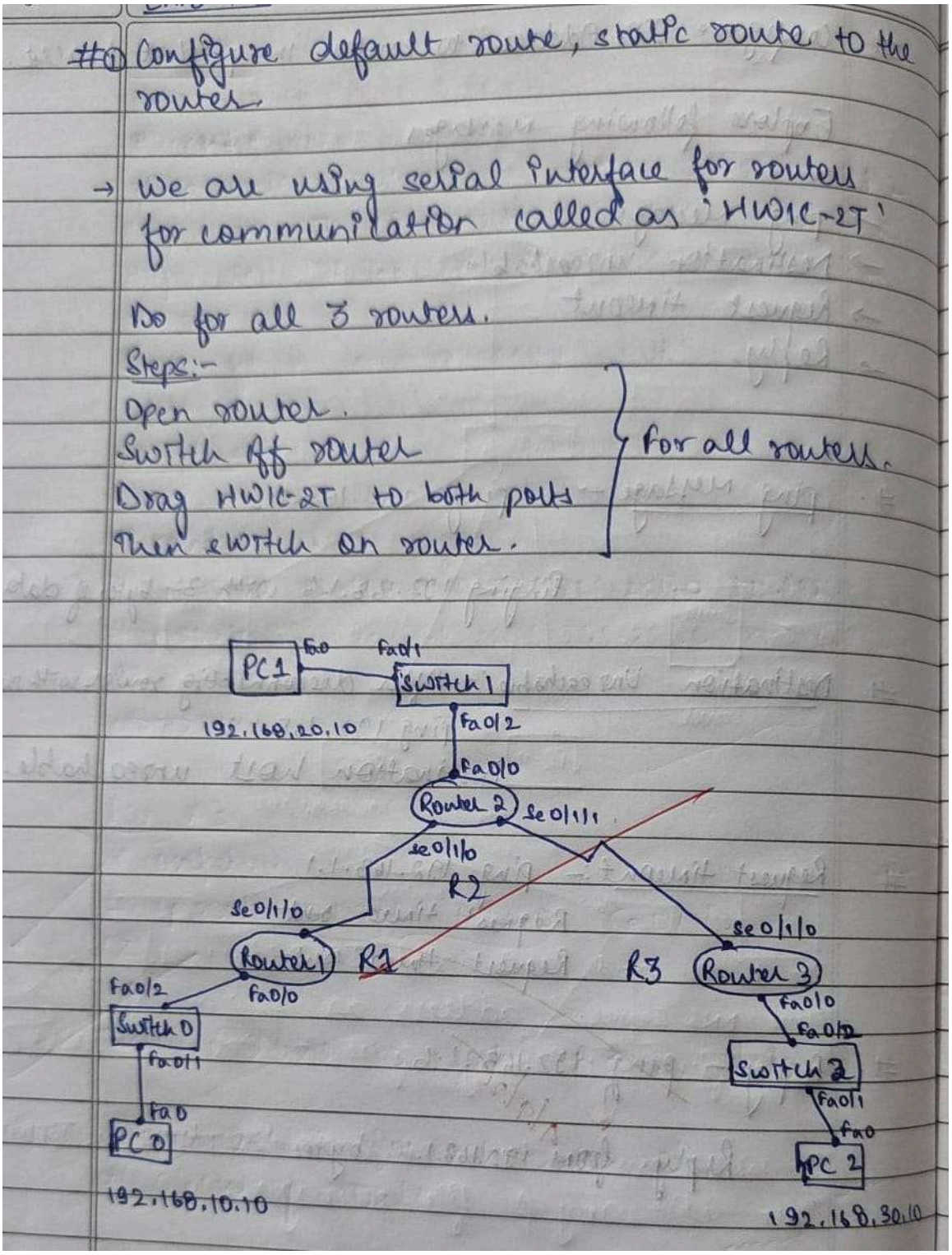
→ We are using serial interface for routers for communication called as 'HWIC-2T'

Do for all 3 routers.

Steps:-

Open router.
Switch off router
Drag HWIC-2T to both ports
Then switch on router.

} for all routers.



Next, in each router's CLI, configure interface for each connection.

R1

enable
conf t
int se0/1/0
ip address 172.16.1.1 255.255.255.252
no shutdown
exit
interface fa0/0
ip address 192.168.10.1 255.255.255.0
no shutdown
write memory.

Similarly, for R2, configure for 3 connections: fa0/0, se0/1/0, se0/1/1

for R3, configure for 2 connections: fa0/0, se0/1/0.

Next, configure IP addresses for all 3 PC's.

For PC0.

IP: 192.168.10.10

Default Gateway: 192.168.10.0

Same for rest PCs.

Next, in each router's CLI establish router's path.

For R1

conf t.
ip route 192.168.20.0 255.255.255.0 172.16.1.2
ip route 172.16.2.0 255.255.255.252 172.16.1.2
ip route 192.168.30.0 255.255.255.0 172.16.1.2
<ctrl.z>
wr.

For R2

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
^z
wr.

For R3

conf t
ip route 0.0.0.0 0.00.0 Se 0/1/0
^z.
wr
TW, is done to make R3 as default.

Next, in CLI for each router, do
show ip route

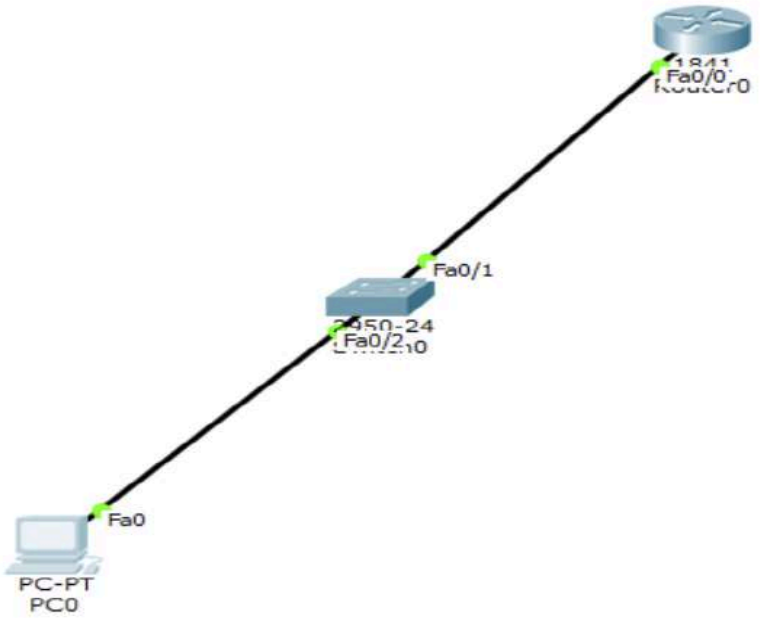
If all successfull, it is right.

Next in each PC's command prompt do,
ping 192.168.10.1
192.168.20.1
192.168.30.1
} All successfull.

Program – 5:

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

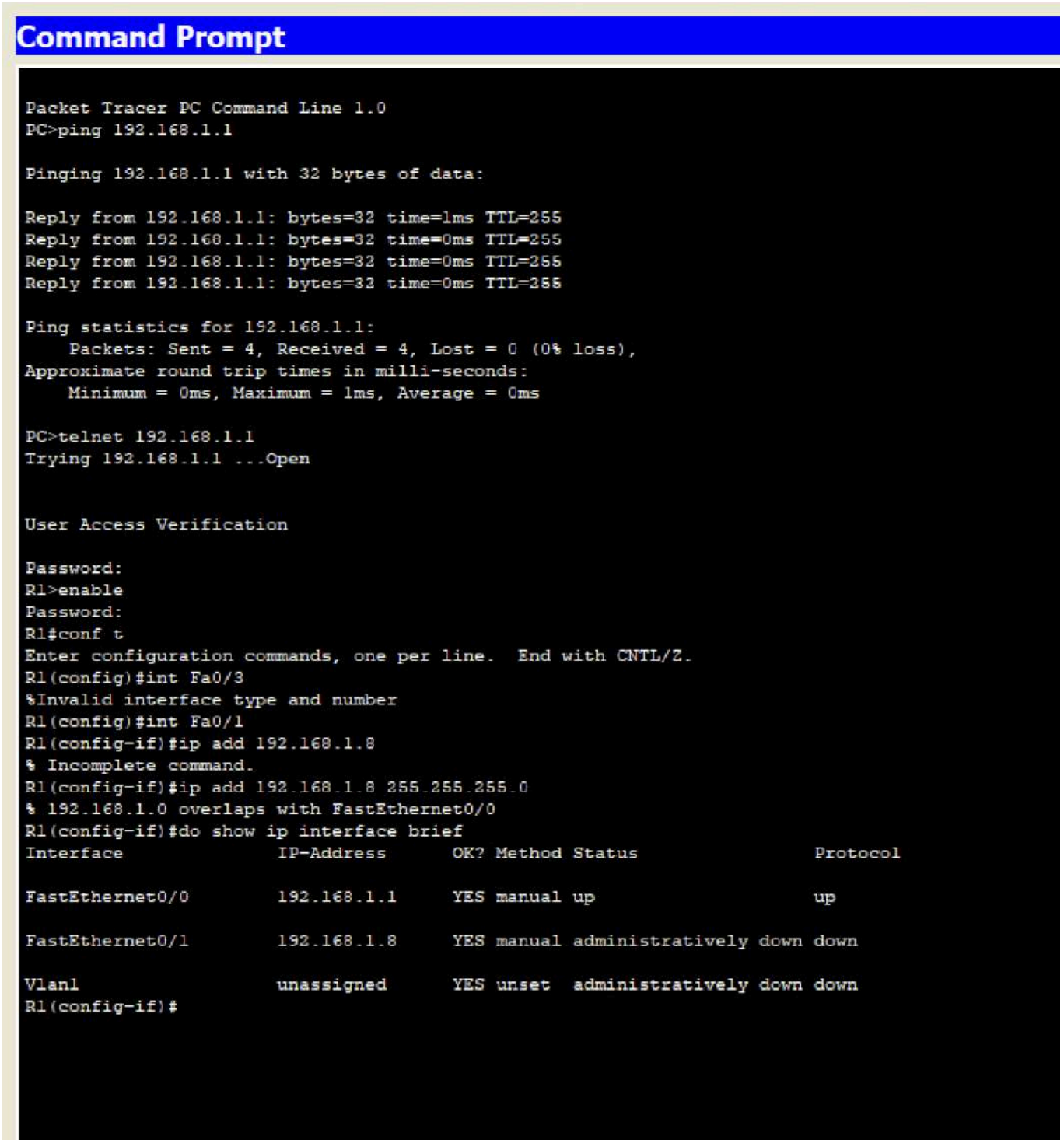
Topology:



Procedure:

1. Configure PC0 : {IP : 192.168.1.2; Default Gateway 192.128.1.1 (router's address)}
2. Configure Router:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# hostname R1
    d. R1(config)# enable secret rp (rp - variable (password for enable))
    e. R1(config)# int Fa0/0
    f. R1(config-if)# ip address 192.168.1.1 255.255.255.0
    g. R1(config-if)# line vty 0 5
    h. R1(config-line)# login
    i. R1(config-line)# password tp (tp - variable password)
    j. R1(config-line)# exit
    k. R1(config)# exit
    l. Write memory
    m. R1# show ip address brief
3. Now go in command prompt of PC:
    a. Ping 192.168.1.1 (to check connection - successfull)
    b. telnet 192.168.1.1
    c. Password : tp
    d. enable
    e. Password : rp
    f. conf t
    g. int Fa0/1
    h. ip address 192.168.1.4 255.255.255.0
    i. do show ip address brief

Output


Command Prompt
Packet Tracer PC Command Line 1.0
PC>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=0ms TTL=255
Reply from 192.168.1.1: bytes=32 time=0ms TTL=255
Reply from 192.168.1.1: bytes=32 time=0ms 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 = 1ms, Average = 0ms
PC>telnet 192.168.1.1
Trying 192.168.1.1 ...Open
User Access Verification
Password:
R1>enable
Password:
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#int Fa0/3
%Invalid interface type and number
R1(config)#int Fa0/1
R1(config-if)#ip add 192.168.1.8
% Incomplete command.
R1(config-if)#ip add 192.168.1.8 255.255.255.0
% 192.168.1.0 overlaps with FastEthernet0/0
R1(config-if)#do show ip interface brief
Interface IP-Address OK? Method Status Protocol
FastEthernet0/0 192.168.1.1 YES manual up up
FastEthernet0/1 192.168.1.8 YES manual administratively down down
Vlan1 unassigned YES unset administratively down down
R1(config-if)#

Observation:

Configure Telnet to access router remotely.

→ Telnet is used to access remote server / router & its a simple command line tool that runs on your computer & it allows to send commands remotely through a server & administrator.

→ Telnet is also used to manage other devices like router, switch & also to check if the ports are open / close.

Create topology with 1 router, 1 switch, 1 PC.

Configure PC address 192.168.1.2

Next in router's CLI :-

```
no
enable
conf t
hostname R1
enable secret rp                    (# rp → variable name)
int fa0/0                           (interface from router)
ip address 192.168.1.1 255.255.255.0   (router add)
no shutdown

line vty 0 5                        (virtual interface with bandwid
login

password tp
exit
```

exit

or

* show ip address brief.

Next, go in command prompt of PC.

- ping 192.168.1.1 (router's ip to check connection)

* if successful >

- telnet 192.168.1.1

<Vtel Access verification

Password : tp

R1> enable

Password : tp

R1# conf t

int fa0/1

ip add 192.168.1.4 255.255.255.0

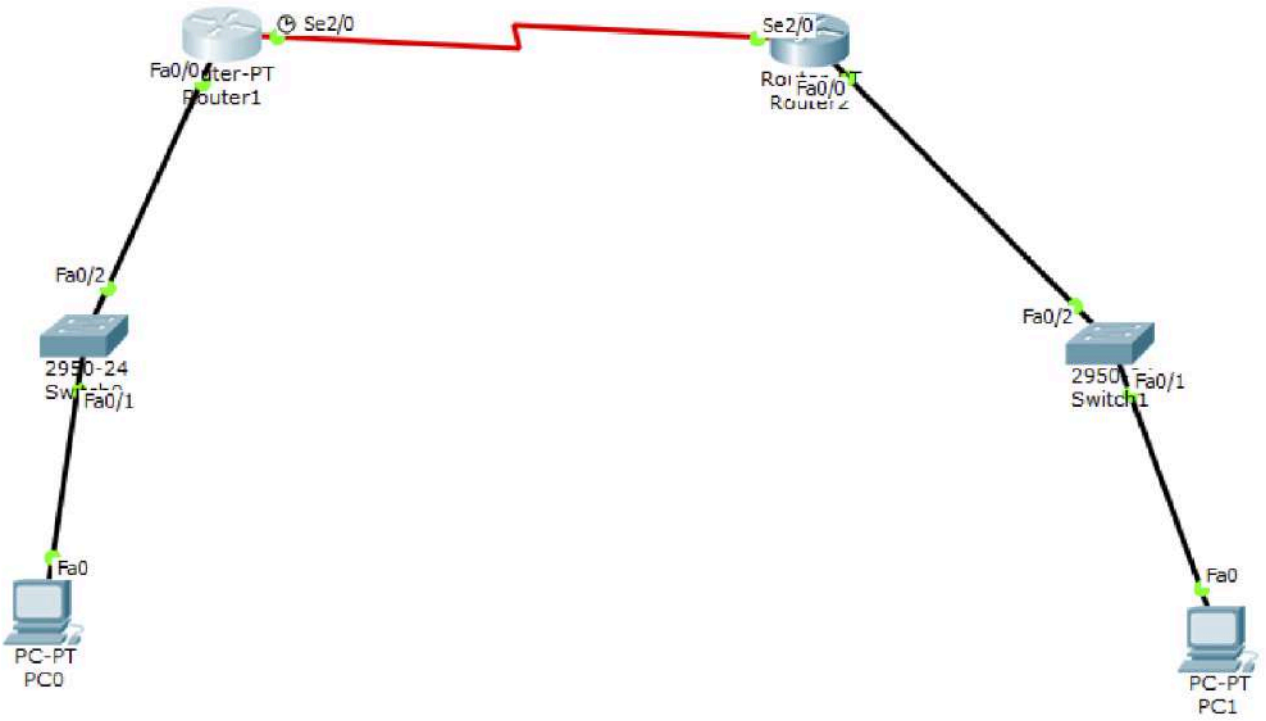
* do show ip interface brief.

Next add 2 more PCs. then update fa0/1 through those PC's command prompt.

Program – 6:

Aim: Configure RIP routing Protocol in Routers

Topology:



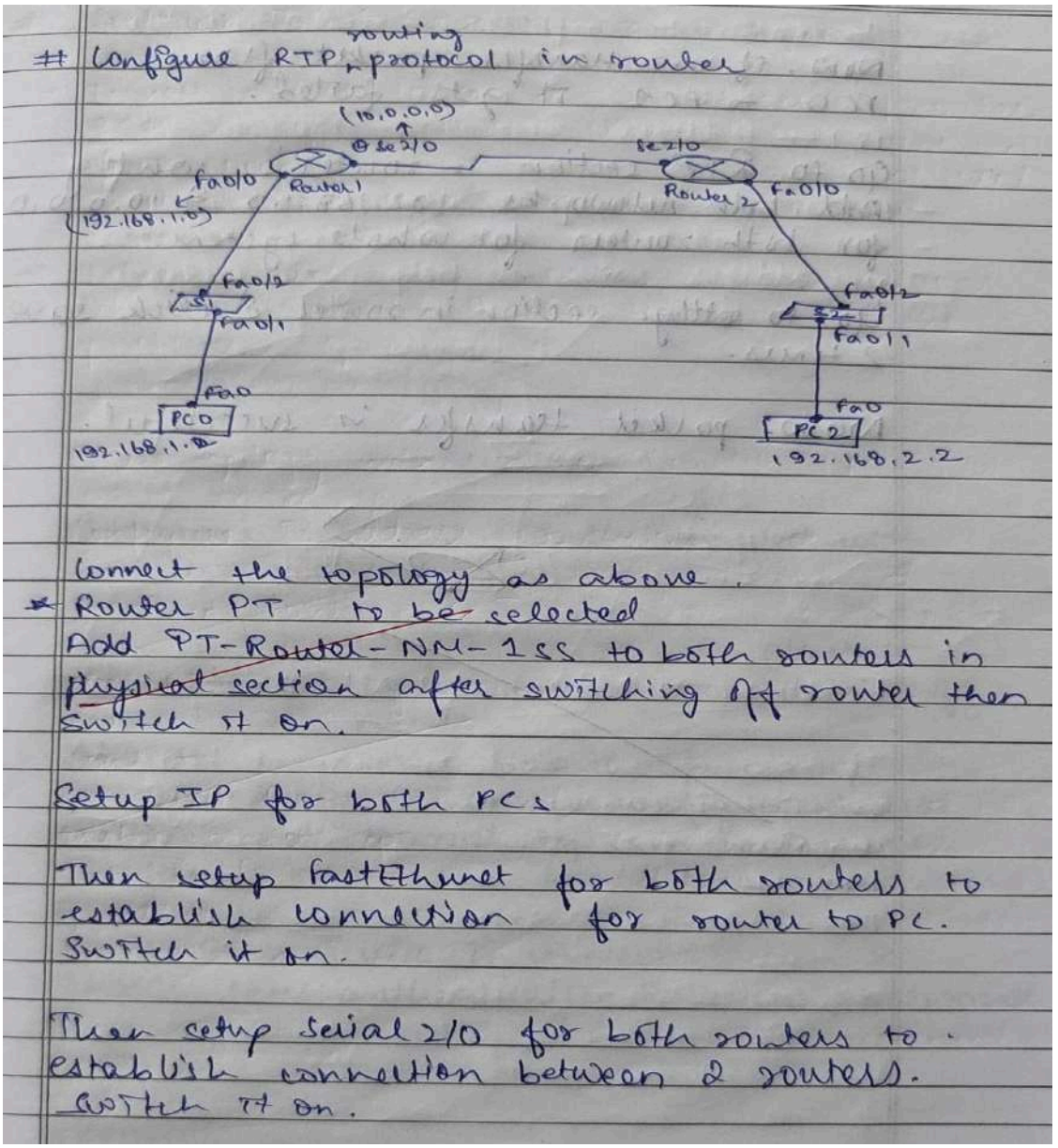
Procedure:

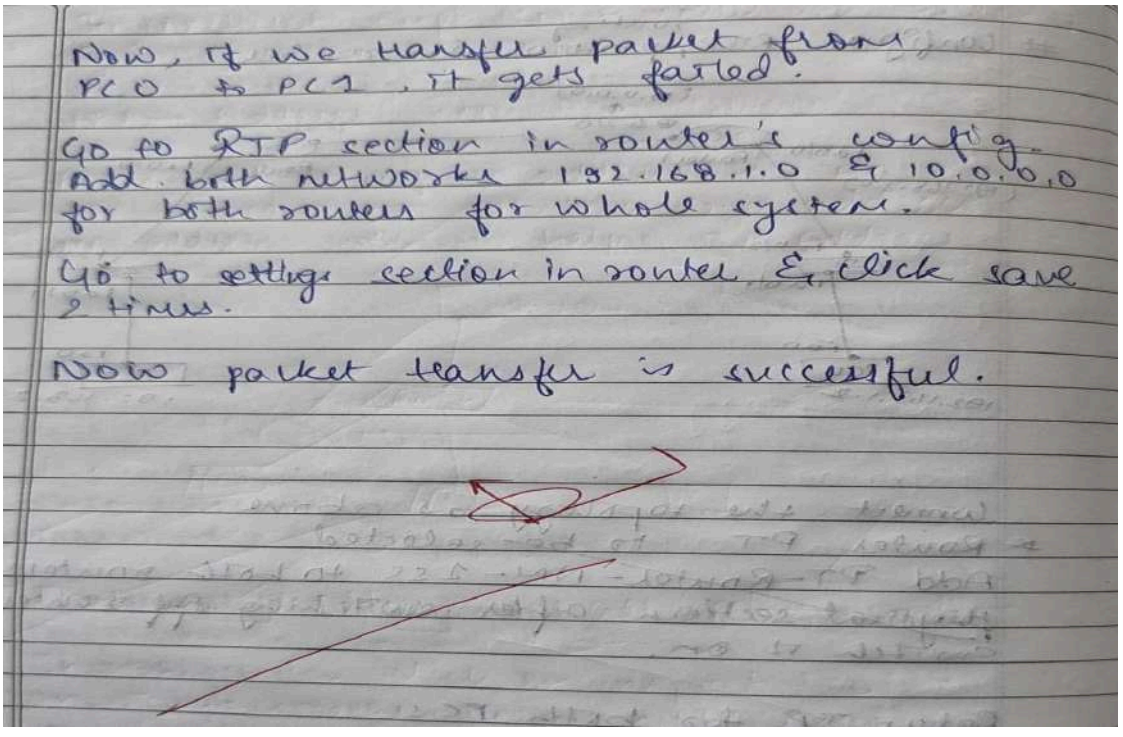
1. Use **Router-PT** in this
2. Configure PC0 {IP : 192.168.1.2, Default Gateway: 192.168.1.1}; PC1 {IP: 192.168.2.2, Default Gateway: 192.168.2.1}
3. In both router Switch off the router → Add "Router-PT-NM-1SS" to both slots → Switch on the router
4. Setup FastEthernet (Fa0/0) for both routers : {IP1 : 192.168.1.1; IP2 : 192.168.2.1}; Tx range limit : 10
5. Setup Serial2/0 (Se2/0) for both routers (IP : 10.0.0.2); Tx range limit : 10
6. Now if packet is passed from PC0 to PC1 → FAILS
7. Go to Router → config → RIP routing → Add both 192.168.1.0 and 10.0.0.0 for both routers
8. Go to Router → config → Settings → Save 2 times

Output :

Packet transfer from PC0 to PC1 is successful.

Observation:

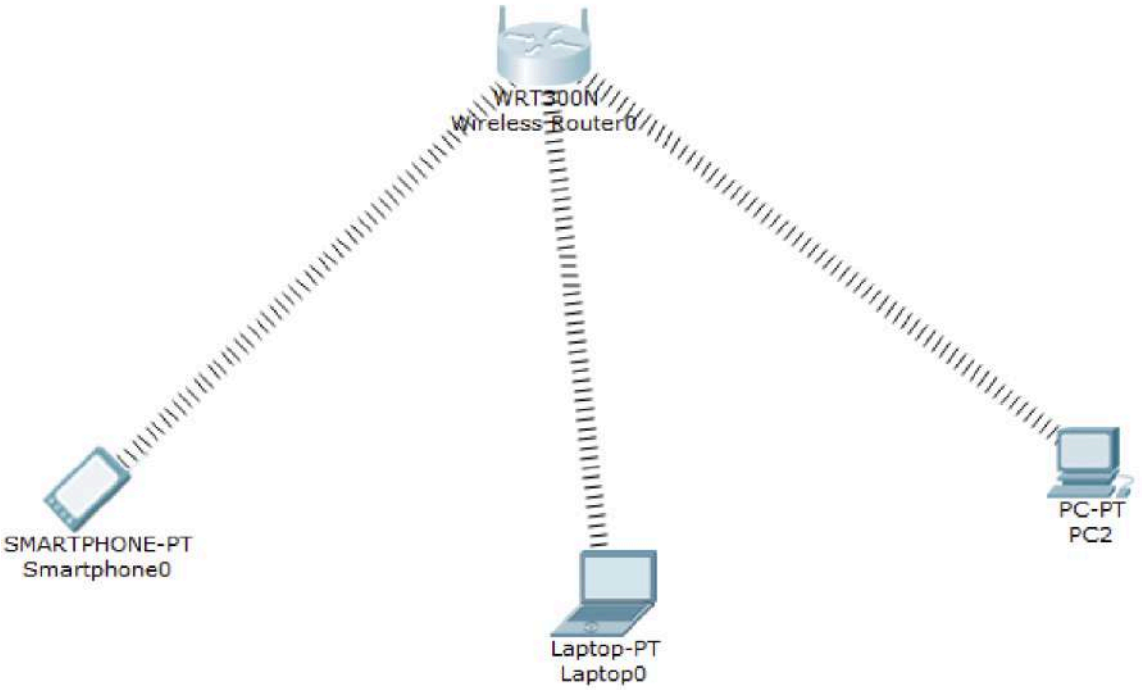




Program – 7:

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

Topology:



Procedure:

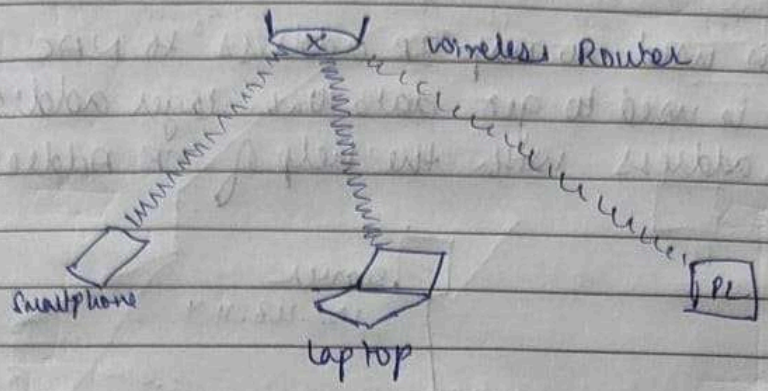
1. Use 3 different devices : Smartphone, Laptop and PC
2. Smartphone gets automatically connected to the wireless router
3. In laptop , remove ethernet cable from physical structure after switching it off and add WPC300N instead, then switch it ON.
4. In PC , remove ethernet cable from physical structure after switching it off and add WMP300N instead, then switch it ON.
5. All devices are now connected with OPEN WiFi.
6. For CLOSED WiFi, go in router config → Settings → set SSID to ‘BMSCE’ nd authentication WPA-2 PSK and sets it’s WPS Key : ‘BMSCE12345’.
7. Next go in each device’s config → Wireless() → give same SSID and Authentication as in router.

Output:

All devices are connected to Closed Wifi

Observation:

To construct a WLAN & make the nodes communicate wirelessly.



Smallphone gets automatically connected with wireless router

Then in both laptop & PC remove ethernet cable & add WPC300N after switching off Then switch on

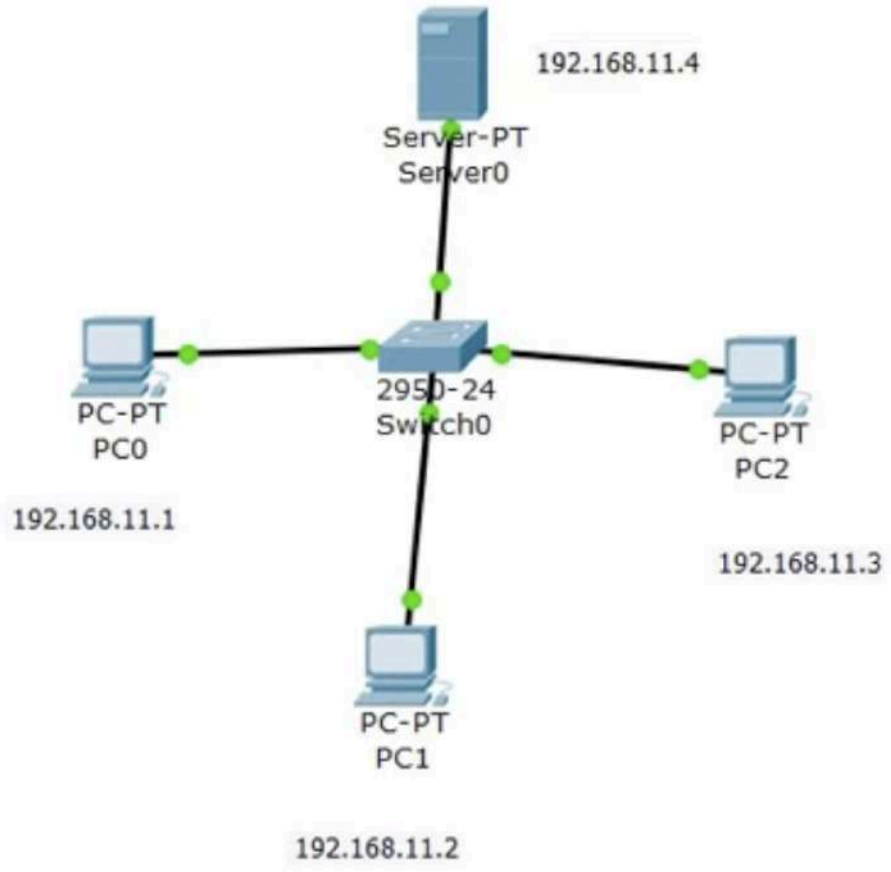
All are connected but to open wifi for that, go in router config in wireless section. Set SSID to BMSCE & select WPA 2-PSK & add Phrase (password).

Go in each device in wireless section & choose same authentication & SSID, password.

Program – 8:

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

Topology:



Procedure:

1. Configure IP Addresses of all PCs and Server as in figure (NO Default Gateway)
2. Select Resolution option
3. Click on Inspect Button(top right side of screen) → Click on PC0 and Server → Select ARP Table from each. Initially it would be empty.
4. Go to command prompt of PC0:
    a. ARP -A
    b. // No ARP entries found
    c. Ping 192.168.11.4 (wherever the message has to be sent)
5. We see packet transferring and ARP tables getting filled in both.

Output:

ARP Table for Server0

IP Address	Hardware Address	Interface
192.168.11.1	0060.47B9.BDE3	FastEthernet0

ARP Table for PC0

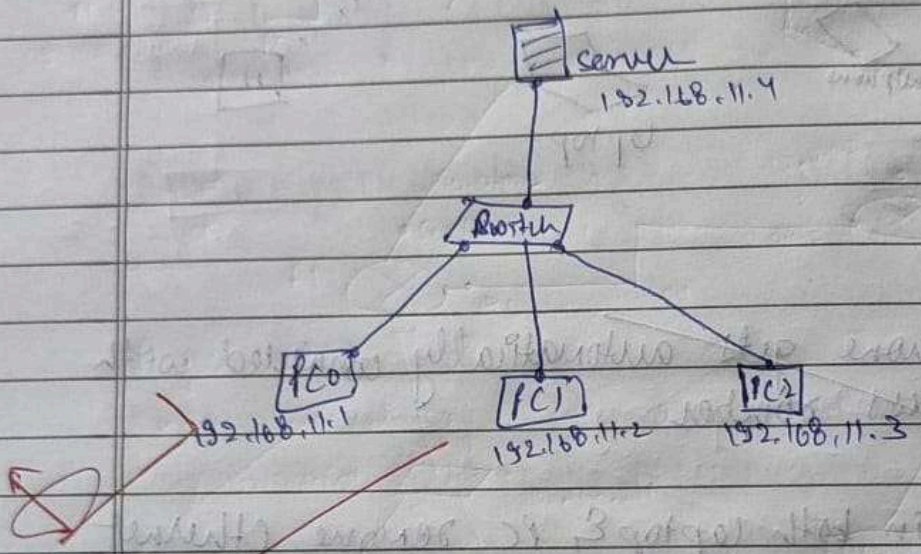
IP Address	Hardware Address	Interface
192.168.11.4	0000.0C13.E51D	FastEthernet0

Observation:

To construct simple LAN & understand the concept & operation of ARP.

ARP:- Address Resolution Protocol.

- ARP is used to map IP address to MAC
- ARP is used to get data link layer address, MAC address with the help of IP Address.



Establish IP Address for all PCs & server

In right section click on inspect tool (Q). Click on PC0 & server. Then in PC0 go to command prompt:

ARP -A

PING 192.168.11.1

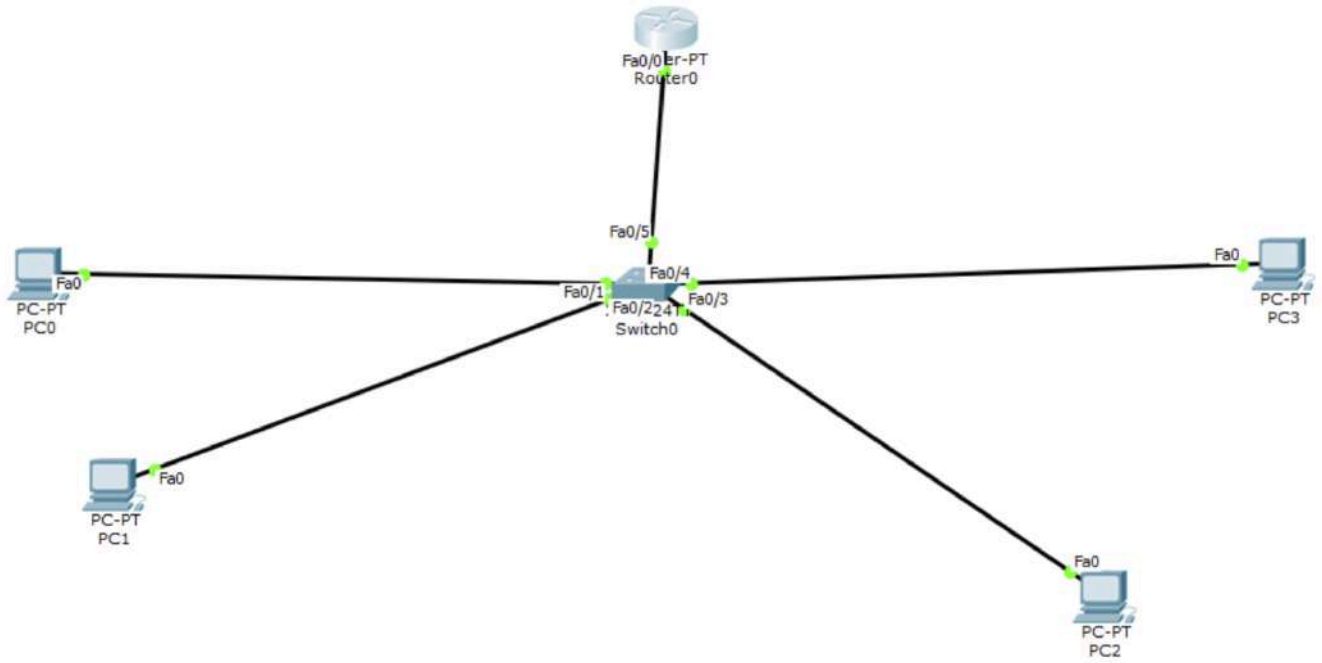
We see entries in ARP.

See simulation of packet from PC0 to server

Program – 9:

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

Topology:

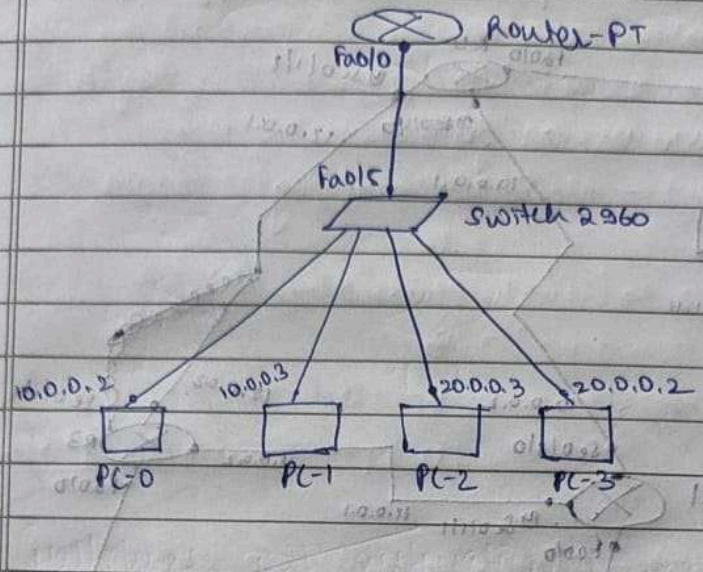


Procedure:

1. Setup Router's IP Address for Fast Ethernet0/0 (Fa0/0) : 10.0.0.1
2. Next Configure PC0 and PC1. In Fa0 do PC0{IP : 10.0.0.2, Gateway : 10.0.0.1}; PC1{IP : 10.0.0.3, Gateway : 10.0.0.1}
3. Next Configure PC2 and PC3. In Fa0 do PC2{IP : 20.0.0.3, Gateway : 20.0.0.1}; PC1{IP : 20.0.0.3, Gateway : 20.0.0.1} [Different VLAN]
4. Next go to Switch Config → Fa0/5 → Select 'Trunk'
5. Next go to Switch Config → VLAN Database → Add new VLAN with name and number (vlan,2)
6. Configure Router's CLI
    a. Router>enable
    b. Router#conf t
    c. Router(config)# int Fa0/0
    d. Router(config-if)# ip address 10.0.0.1 255.0.0.0
    e. Router(config-if)# no shutdown
    f. Router(config-if)# exit
    g. Router(config)# int Fa0/0.1
    h. Router(config-subif)# encapsulation dot1q 2
    i. Router(config-subif)# ip address 20.0.0.1 255.0.0.0
    j. Router(config-subif)# no shutdown
    k. Router(config-subif)# exit
    l. Router(config)# exit
    m. Write memory

Observation:

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



first set up id address for router's Fa0/0 to 10.0.0.1.

Next set up PC0 & PC1 :- Both gateways 10.0.0.1
In FastEthernet0, ip address :- 10.0.0.2 & 10.0.0.3.

Set up PC2 & PC3 :- Both gateways 20.0.0.1 (diff. VLAN)
In F0, ip add :- 20.0.0.3 & 20.0.0.2.

Next go in switch config, open fa0/5, select "Trunk" (manage traffic b/w different VLANs).

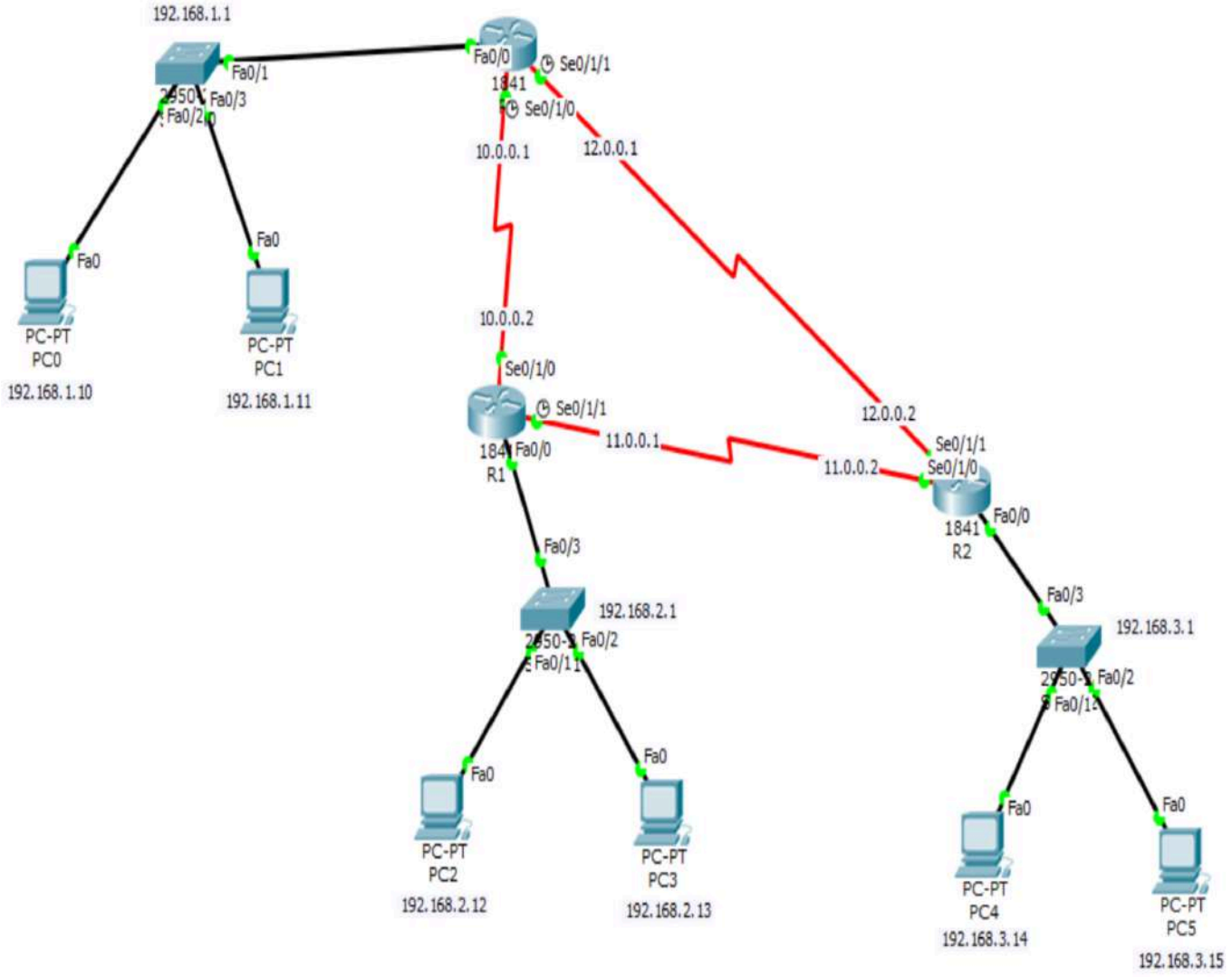
Next go to VLAN Database in switch, add new VLAN with number 2 & name vlan.

Next update router's CLI.

Program – 10:

Aim: Configure OSPF routing protocol

Topology:



Procedure:

1. In each router, switch off → Add 'HWIC-2T' in both ports → switch on
2. Configure all PCs according to diagram
3. Configure R0:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# hostname R0
    d. R0(config)# int Fa0/0
    e. R0(config-if)# ip address 192.168.1.1 255.255.255.0
    f. R0(config-if)# no shutdown
    g. R0(config-if)# exit
    h. R0(config)# int Se0/1/0
    i. R0(config-if)# ip address 10.0.0.1 255.0.0.0
    j. R0(config-if)# clock rate 64000
    k. R0(config-if)# no shutdown
    l. R0(config-if)# exit

m. R0(config)# int Se0/1/1
n. R0(config-if)# ip address 12.0.0.1 255.0.0.0
o. R0(config-if)# clock rate 64000
p. R0(config-if)# no shutdown
q. R0(config-if)# exit
r. R0(config)# router ospf 1
s. R0(config-router)# network 192.168.1.0 0.0.0.255 area 0
t. R0(config-router)# network 10.0.0.0 0.255.255.255 area 0
u. R0(config-router)# network 12.0.0.0 0.255.255.255 area 0
v. R0(config-router)# exit
w. R0(config)# exit
x. wr

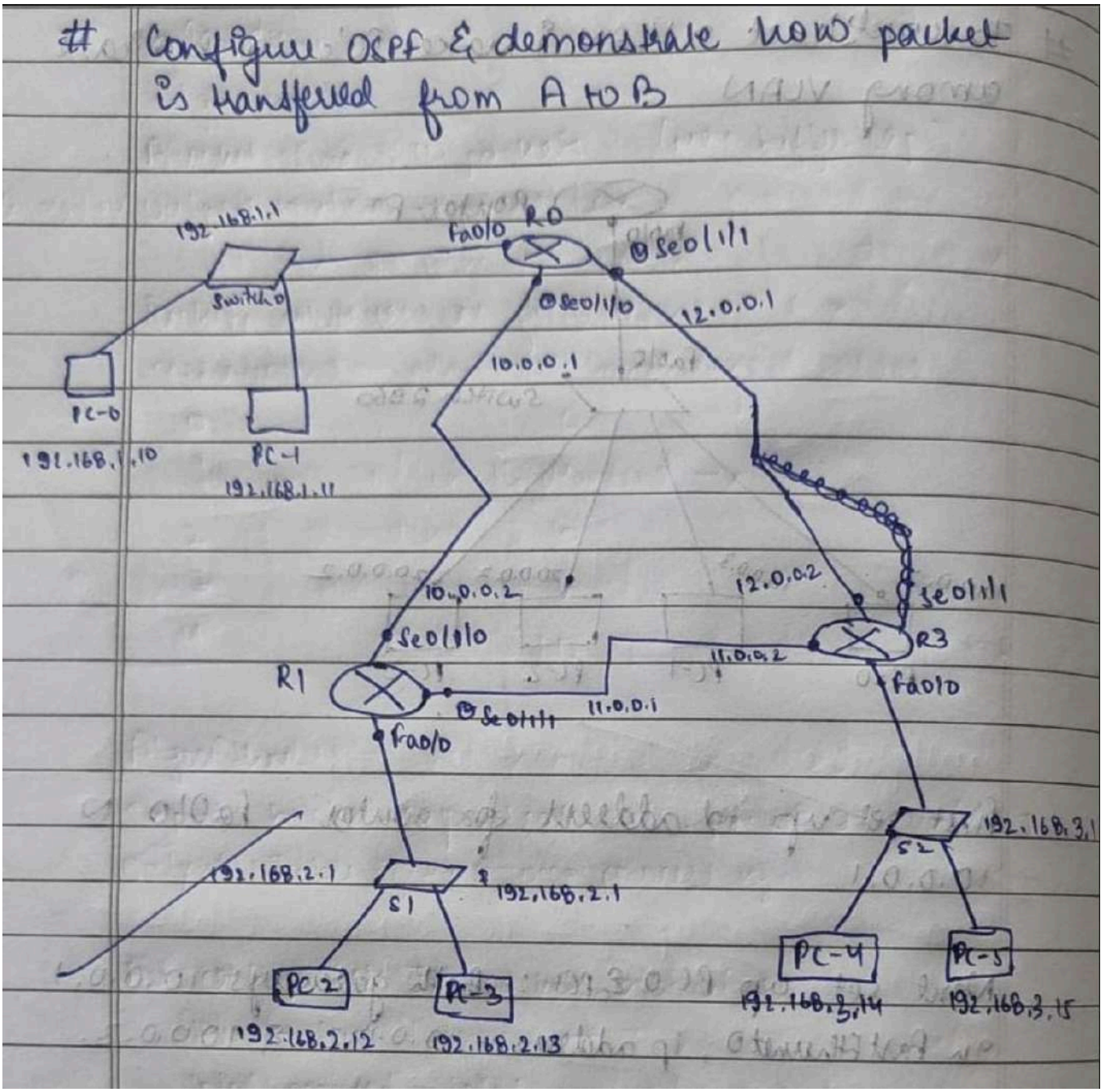
4. Configure R1:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# hostname R1
    d. R1(config)# int Fa0/0
    e. R1(config-if)# ip address 192.168.2.1 255.255.255.0
    f. R1(config-if)# no shutdown
    g. R1(config-if)# exit
    h. R1(config)# int Se0/1/0
    i. R1(config-if)# ip address 10.0.0.2 255.0.0.0
    j. R1(config-if)# clock rate 64000
    k. R1(config-if)# no shutdown
    l. R1(config-if)# exit
    m. R1(config)# int Se0/1/1
    n. R1(config-if)# ip address 11.0.0.1 255.0.0.0
    o. R1(config-if)# clock rate 64000
    p. R1(config-if)# no shutdown
    q. R1(config-if)# exit
    r. R1(config)# router ospf 1
    s. R1(config-router)# network 192.168.2.0 0.0.0.255 area 0
    t. R1(config-router)# network 10.0.0.0 0.255.255.255 area 0
    u. R1(config-router)# network 11.0.0.0 0.255.255.255 area 0
    v. R1(config-router)# exit
    w. R1(config)# exit
    x. wr
5. Configure R2:
    a. Router>enable
    b. Router#conf t
    c. Router(config)# hostname R2
    d. R2(config)# int Fa0/0
    e. R2(config-if)# ip address 192.168.3.1 255.255.255.0
    f. R2(config-if)# no shutdown
    g. R2(config-if)# exit
    h. R2(config)# int Se0/1/0
    i. R2(config-if)# ip address 11.0.0.2 255.0.0.0
    j. R2(config-if)# clock rate 64000
    k. R2(config-if)# no shutdown
    l. R2(config-if)# exit
    m. R2(config)# int Se0/1/1
    n. R2(config-if)# ip address 12.0.0.2 255.0.0.0
    o. R2(config-if)# clock rate 64000
    p. R2(config-if)# no shutdown

q. R2(config-if)# exit
r. R2(config)# router ospf 1
s. R2(config-router)# network 192.168.3.0 0.0.0.255 area 0
t. R2(config-router)# network 11.0.0.0 0.255.255.255 area 0
u. R2(config-router)# network 12.0.0.0 0.255.255.255 area 0
v. R2(config-router)# exit
w. R2(config)# exit
x. wr

Output:

Fire	Last Status	Source	Destination	Type	Color	Time(sec)	Periodic	Num	Edit	Delete
	Successful	PC1	PC3	ICMP		0.000	N	2	(edit)	(delete)
	Successful	PC4	PC3	ICMP		0.000	N	3	(edit)	(delete)
	Successful	PC5	PC1	ICMP		0.000	N	4	(edit)	(delete)
	Successful	PC0	PC5	ICMP		0.000	N	5	(edit)	(delete)

Observation:



Firstly, go in each router switch PT off & add 'HWIC-2T' ports in both spaces & then switch it on.

Then configure ip address for all PC's.

Then go in CLI in each router.

first, configure interface with its switch then for rest 2 connections made with corresponding 2 routers.

Now,
give command > router ospf 1

Add all 3 networks (3 connections made)

To check ospf, give command as>

show ip route

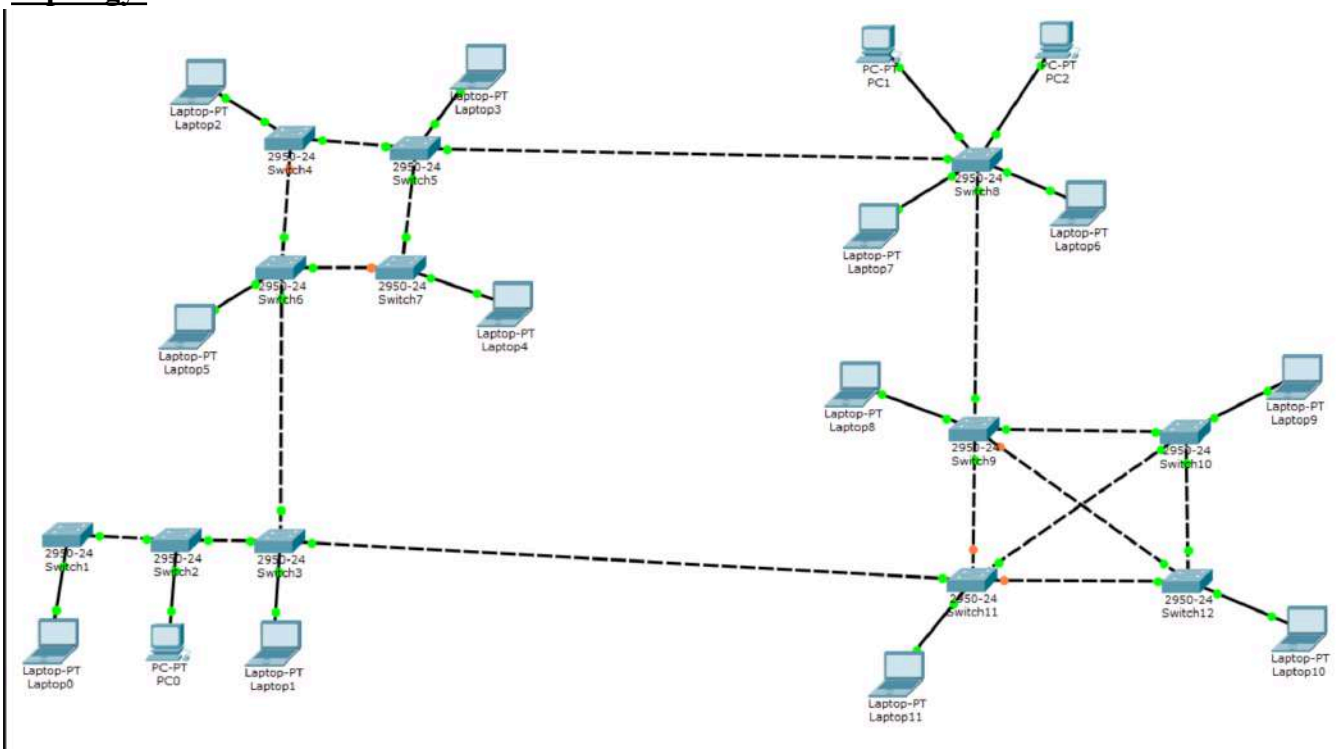
'O' → ospf route.

Then send packet from PC-0 to PC-2 & PC-5 to check connection.

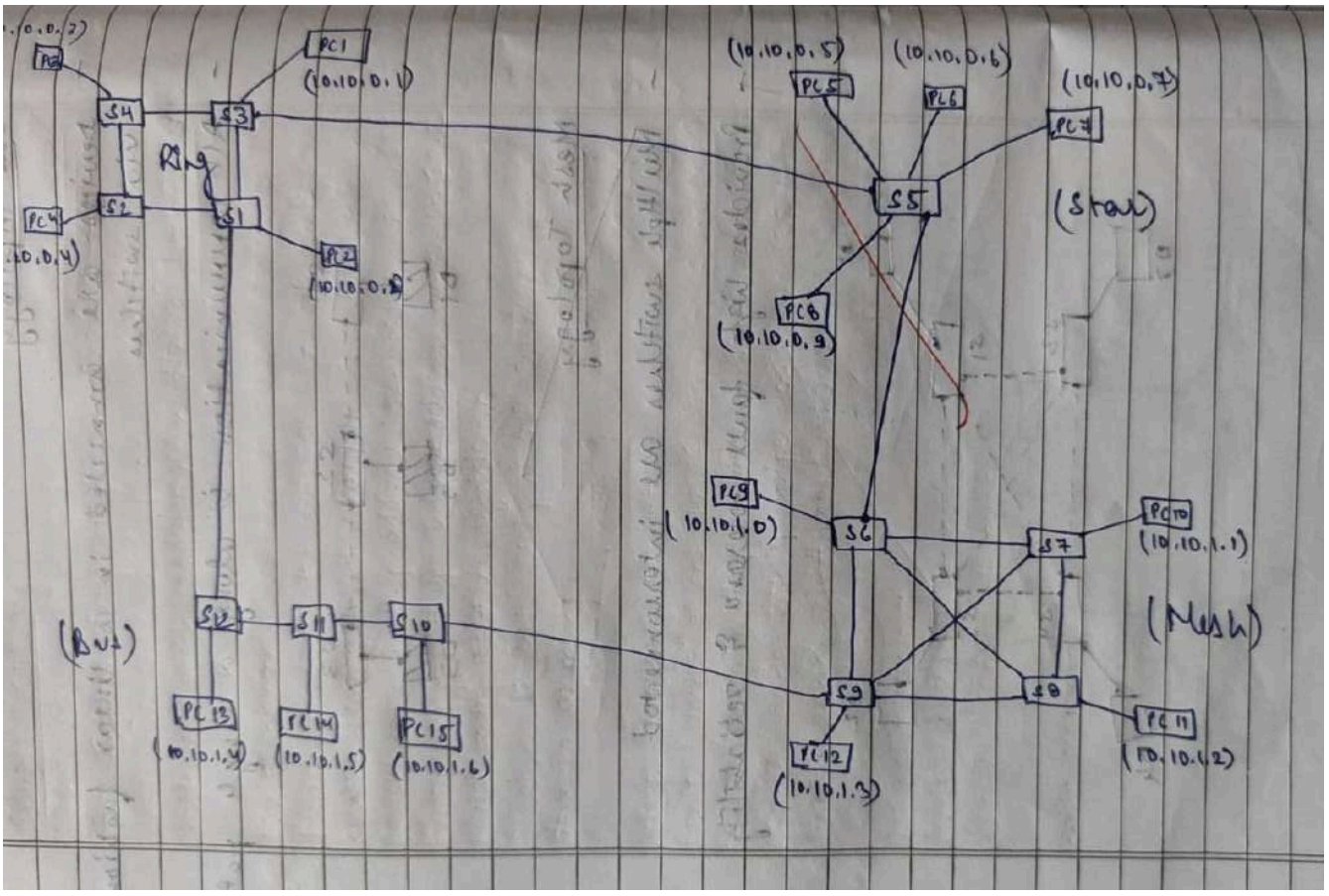
Program – 11:

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:



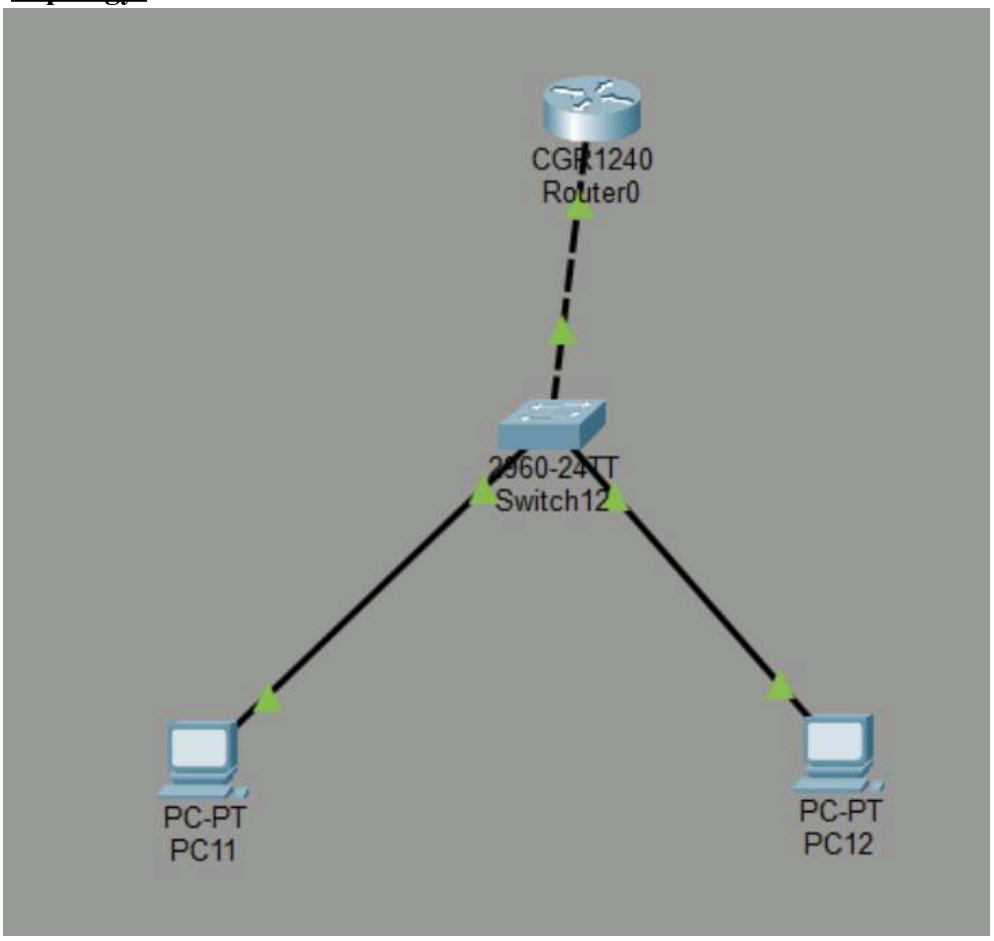
Observation:



Program – 12:

Aim: Demonstrate the TTL/ Life of a Packet

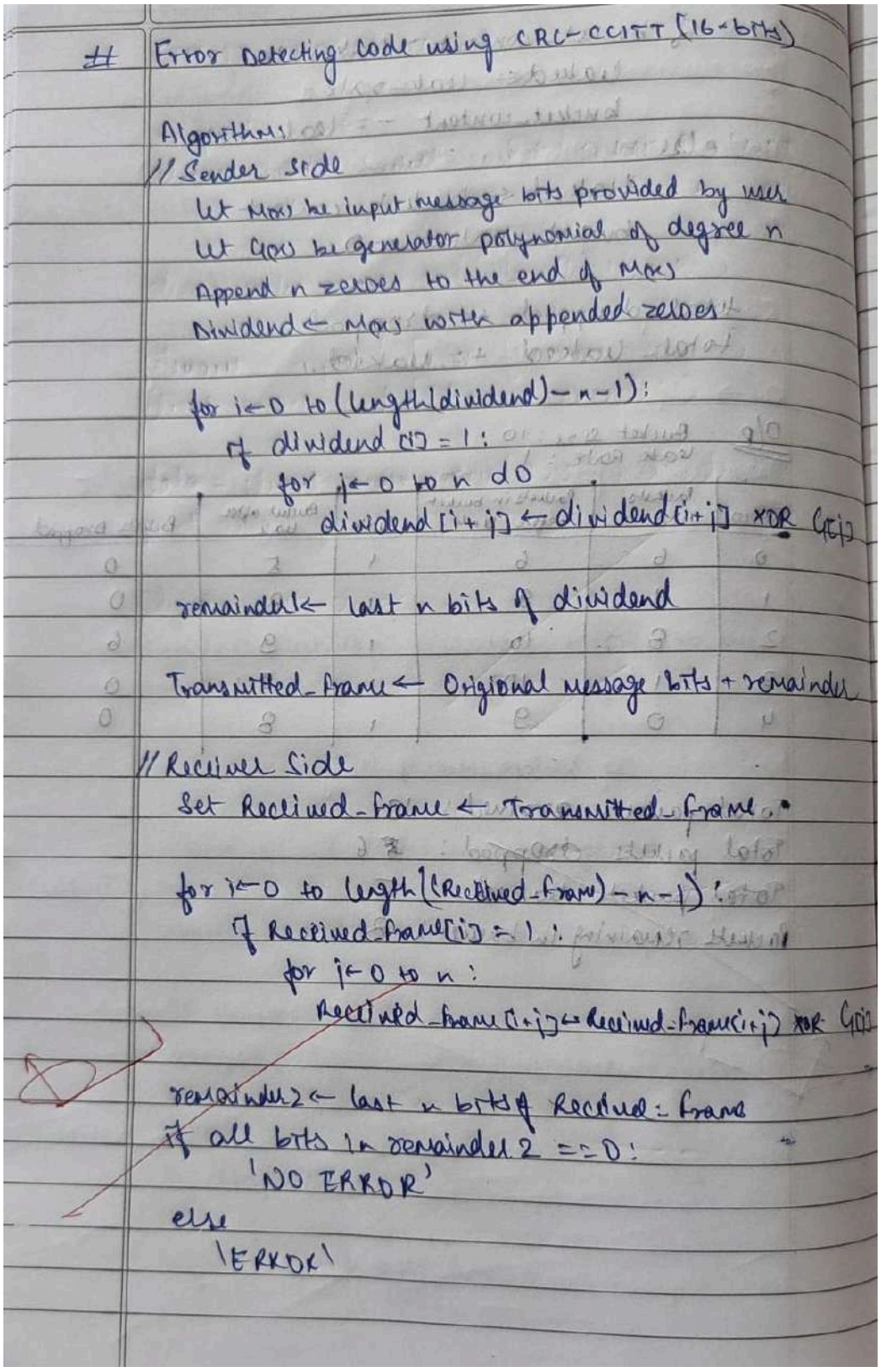
Topology:




```
C:\>ping 192.168.10.3

Pinging 192.168.10.3 with 32 bytes of data:

Reply from 192.168.10.3: bytes=32 time<1ms TTL=128
Reply from 192.168.10.3: bytes=32 time<1ms TTL=128
Reply from 192.168.10.3: bytes=32 time<1ms TTL=128
Reply from 192.168.10.3: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.10.3:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Program – 13:

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



Error Detecting code using CRC-CCITT (16-bits)

Algorithm:

// Sender side

Let Msg be input message bits provided by user
Let Gen be generator polynomial of degree n
Append n zeroes to the end of Msg
Dividend ← Msg with appended zeroes

for i ← 0 to (length(dividend) − n − 1):
    if dividend[i] = 1:
        for j ← 0 to n do
            dividend[i+j] ← dividend[i+j] XOR Gen[j]

remainder ← last n bits of dividend

Transmitted_frame ← Original message bits + remainder

// Reciver Side

Set Recived_frame ← Transmitted_frame

for i ← 0 to (length(Recived_frame) − n − 1):
    if Received_frame[i] = 1:
        for j ← 0 to n:
            Received_frame[i+j] ← Received_frame[i+j] XOR Gen[j]

remainder2 ← last n bits of Received frame
if all bits in remainder 2 == 0:
    'NO ERROR'
else
    'ERROR'

Code:

```
def crc_error_detection(message_bits, generator_bits):
    n = len(generator_bits) - 1  # Degree of polynomial

    # Step 1: Append n zeros to message
    dividend = message_bits + [0] * n

    # Show dividend after adding zeros
    print("Dividend after appending zeros:", dividend)

    # Step 2: Perform Modulo-2 Division (XOR)
    for i in range(len(message_bits)):
        if dividend[i] == 1:
            for j in range(len(generator_bits)):
                dividend[i + j] ^= generator_bits[j]

    # Step 3: Extract remainder (CRC)
    crc = dividend[-n:]

    # Step 4: Form transmitted frame
    transmitted_frame = message_bits + crc

    # ---------------- Receiver Side ----------------
    received_frame = transmitted_frame.copy()

    # Step 5: Repeat division at receiver
    for i in range(len(message_bits)):
        if received_frame[i] == 1:
            for j in range(len(generator_bits)):
                received_frame[i + j] ^= generator_bits[j]

    # Step 6: Check remainder for error
    remainder_check = received_frame[-n:]
    error_detected = any(remainder_check)  # True if any bit is 1

    return {
        "generator": generator_bits,
        "dividend_after_zeros": message_bits + [0] * n,
        "transmitted_frame": transmitted_frame,
        "crc": crc,
        "error_detected": error_detected
    }


# ---------------- User Input ----------------
message_input = input("Enter message bits (e.g., 11010011101100): ")
```

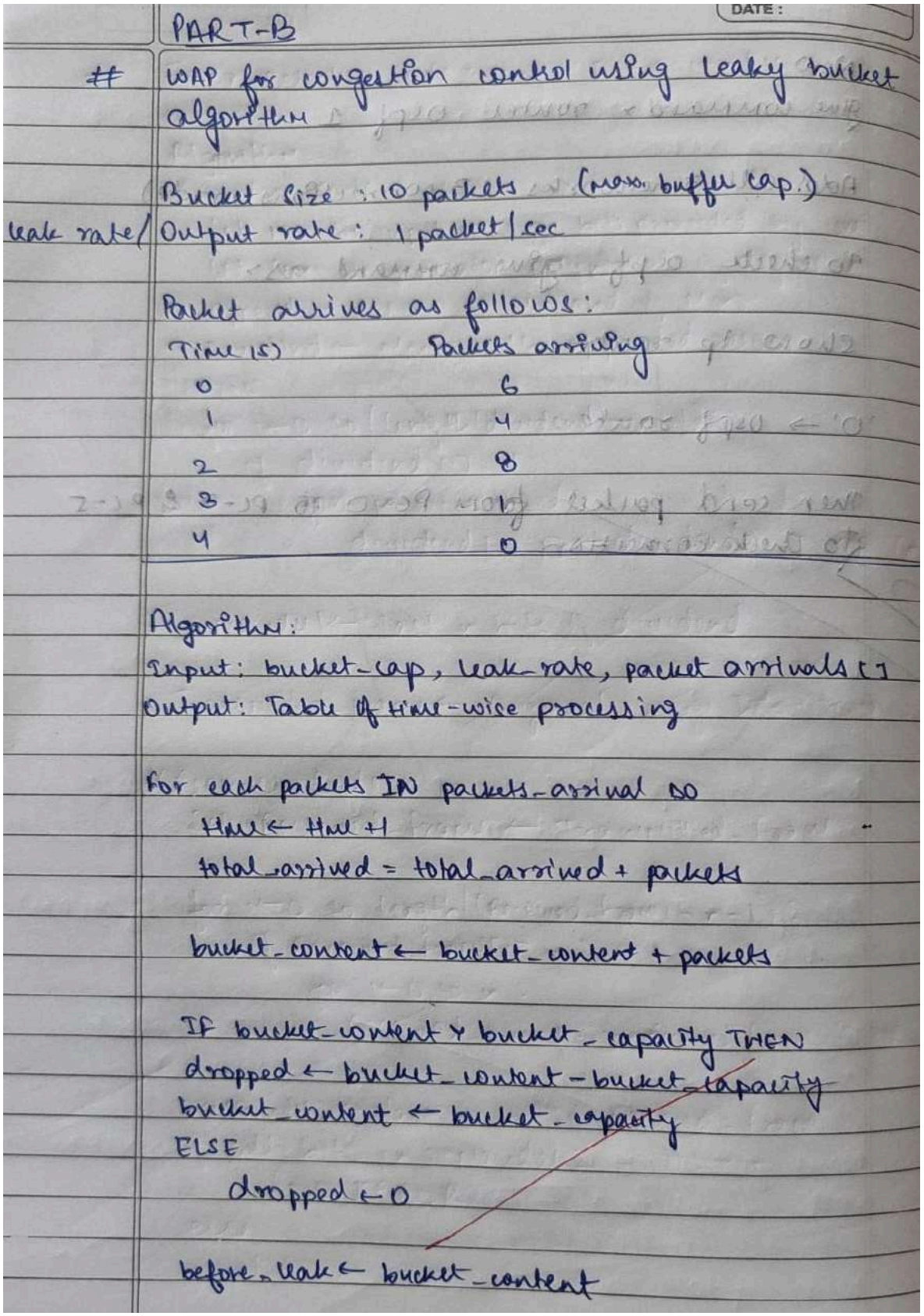
```
generator_input = input("Enter generator polynomial bits (e.g., 10011): ")

# Convert string inputs to list of integers
message_bits = [int(bit) for bit in message_input]
generator_bits = [int(bit) for bit in generator_input]

# Run CRC algorithm
result = crc_error_detection(message_bits, generator_bits)

# ---------------- Output ----------------
print("\nGenerator:", result["generator"])
print("Dividend after adding zeros:", result["dividend_after_zeros"])
print("Transmitted Frame (Message + CRC):", result["transmitted_frame"])
print("CRC Remainder:", result["crc"])
print("Error Detected:", result["error_detected"])
```

Output:

```
Enter the generator polynomial:
101
Generator polynomial (CRC-CCITT): 101
Enter the message:
110101

Message polynomial appended with zeros:
11010100

Checksum (Remainder):
11

Transmitted message (with checksum):
11010111

Enter the received message:
11010111

Received message is error-free ✅


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

Program – 14:

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



PART-B

DATE :

WAP for congestion control using Leaky bucket algorithm

Bucket size : 10 packets (max. buffer cap.)

leak rate / Output rate : 1 packet / sec

Packet arrives as follows:

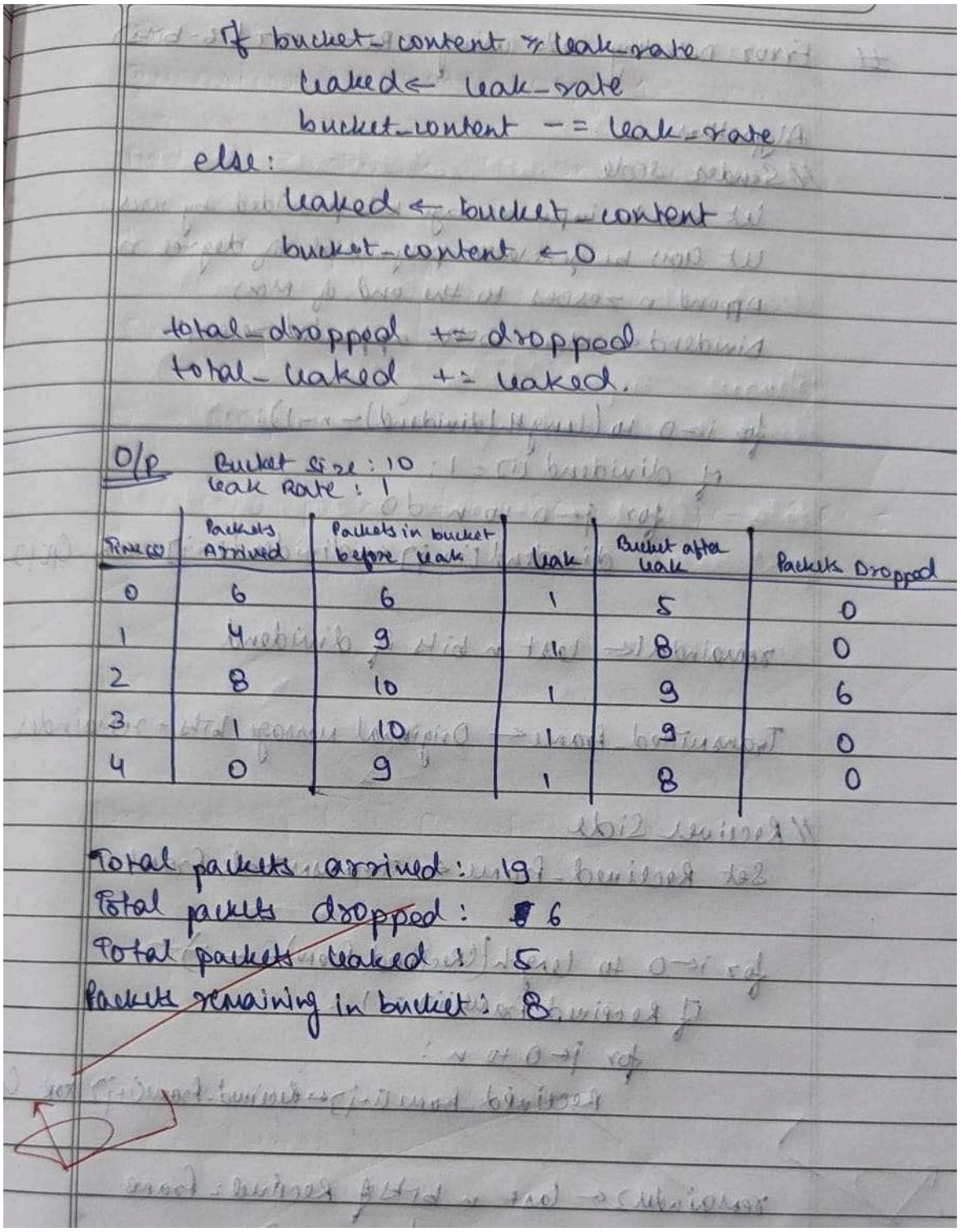
Time (s)	Packets arriving
0	6
1	4
2	8
3	1
4	0

Algorithm:

Input: bucket_cap, leak_rate, packet_arrivals[]

Output: Table of time-wise processing

```
For each packets IN packets_arrival DO
    time ← time + 1
    total_arrived = total_arrived + packets

    bucket_content ← bucket_content + packets

    IF bucket_content > bucket_capacity THEN
    dropped ← bucket_content - bucket_capacity
    bucket_content ← bucket_capacity
    ELSE
        dropped ← 0

    before_leak ← bucket_content
```



Code:

```
def leaky_bucket(leak_rate, bucket_capacity, packet_arrivals):
    time = -1
    bucket_content = 0
    total_arrived = 0       # total packets generated/input
    total_dropped = 0
    total_leaked = 0        # total packets transmitted/output

    # Table header
    print(f"{'Time(s)':<8}{'Packets Arriving':<18}{'Packets in Bucket Before Leak':<32}"
          f"{'Leak':<8}{'Bucket After Leak':<20}{'Packets Dropped':<18}{'Packets in Bucket
Finally':<25}")
```

```
    print("-" * 140)

    # Simulation loop
    for packets in packet_arrivals:
        time += 1
        total_arrived += packets

        # Add incoming packets
        bucket_content += packets

        # Handle overflow
        if bucket_content > bucket_capacity:
            dropped = bucket_content - bucket_capacity
            bucket_content = bucket_capacity
        else:
            dropped = 0

        before_leak = bucket_content  # before leaking

        # Leak packets
        if bucket_content >= leak_rate:
            leaked = leak_rate
            bucket_content -= leak_rate
        else:
            leaked = bucket_content
            bucket_content = 0

        # Print row
        print(f"{time:<8}{packets:<18}{before_leak:<32}{leaked:<8}"
              f"{before_leak - leaked:<20}{dropped:<18}{bucket_content:<25}")

        total_dropped += dropped
        total_leaked += leaked

    # Summary section
    print("\nSummary:")
    print(f"Total Packets Arrived (Produced/Input):   {total_arrived}")
    print(f"Total Packets Leaked (Transmitted/Output): {total_leaked}")
    print(f"Total Packets Dropped:                    {total_dropped}")
    print(f"Packets Remaining in Bucket Finally:      {bucket_content}")

# --- User Input ---
bucket_capacity = int(input("Enter bucket capacity: "))
leak_rate = int(input("Enter leak rate (packets/sec): "))
num_seconds = int(input("Enter number of seconds to simulate: "))

packet_arrivals = []
for i in range(num_seconds):
    pkt = int(input(f"Packets arriving at second {i}: "))  # time starts at 0
    packet_arrivals.append(pkt)

# Run simulation
print("\nLeaky Bucket Simulation:\n")
leaky_bucket(leak_rate, bucket_capacity, packet_arrivals)
```

Output:

```
Enter bucket capacity: 10
Enter leak rate (packets/sec): 1
Enter number of seconds to simulate: 5
Packets arriving at second 0: 6
Packets arriving at second 1: 4
Packets arriving at second 2: 8
Packets arriving at second 3: 1
Packets arriving at second 4: 0

Leaky Bucket Simulation:

Time(s) Packets Arriving  Packets in Bucket Before Leak   Leak    Bucket After Leak   Packets Dropped   Packets in Bucket Finally
-------------------------------------------------------------------------------------------------------------------------------
0       6                 6                             1       5                   0                 5
1       4                 9                             1       8                   0                 8
2       8                 10                            1       9                   6                 9
3       1                 10                            1       9                   0                 9
4       0                 9                             1       8                   0                 8

Summary:
Total Packets Arrived (Produced/Input):   19
Total Packets Leaked (Transmitted/Output): 5
Total Packets Dropped:                    6
Packets Remaining in Bucket Finally:      8
```

Program – 15:

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

```
# TCP Socket Programming

import socket
server_socket = socket.socket(socket.AF_INET,
                              socket.SOCK_STREAM)

server.bind(('localhost', 8080))
server_socket.listen(1)
print("Server is listening on port 8080")

conn, addr = server_socket.accept()
print("Connected by:", addr)

filename = conn.recv(1024).decode()
try:
    with open(filename,'r') as f:
        data = f.read()
    conn.send(data.encode())
except FileNotFoundError:
    conn.send(("File not found"))

conn.close()
server_socket.close()
```

```
CLIENT PROGRAM (TCP)

import socket
client_socket = socket.socket(socket.AF_INET,
                              socket.SOCK_STREAM)

client_socket.connect(('localhost', 8080))

filename = input('Enter filename request')
client_socket.send(filename.encode())

data = client_socket.recv(4096).decode()
print(data)

client_socket.close()
```

Code:

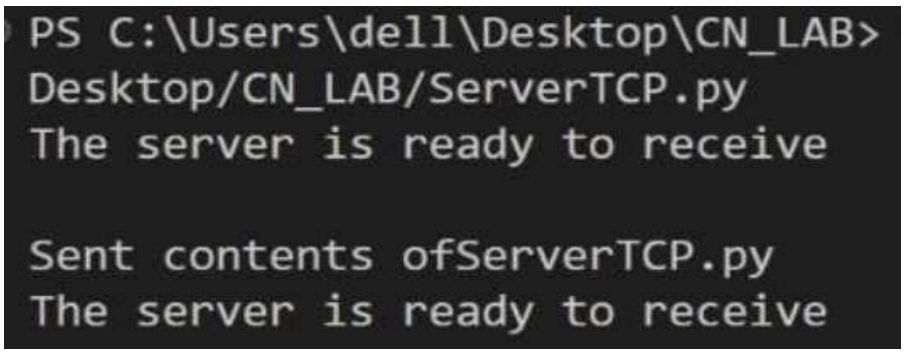
ClientTCP.py

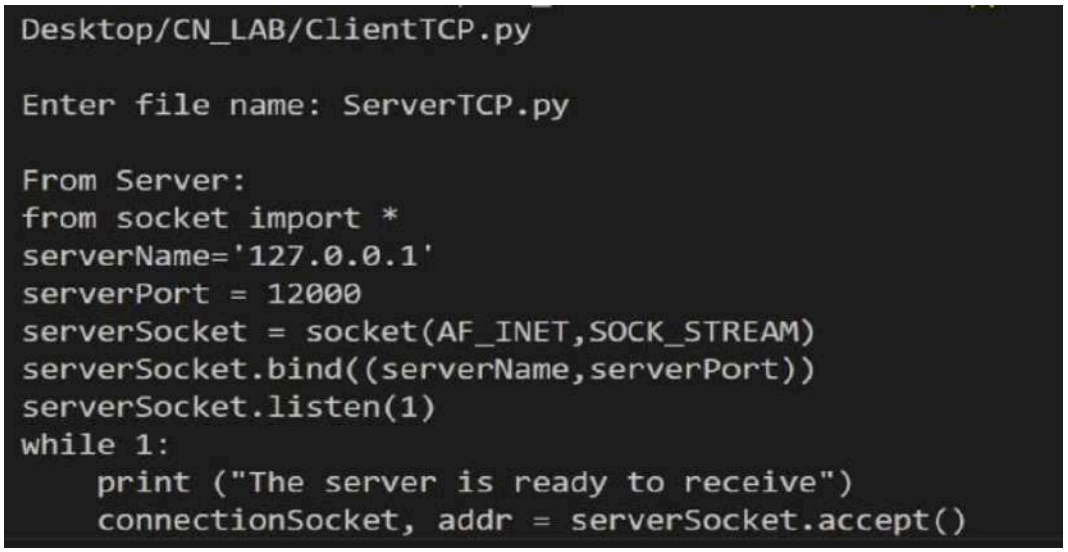
```
from socket import *

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()
```

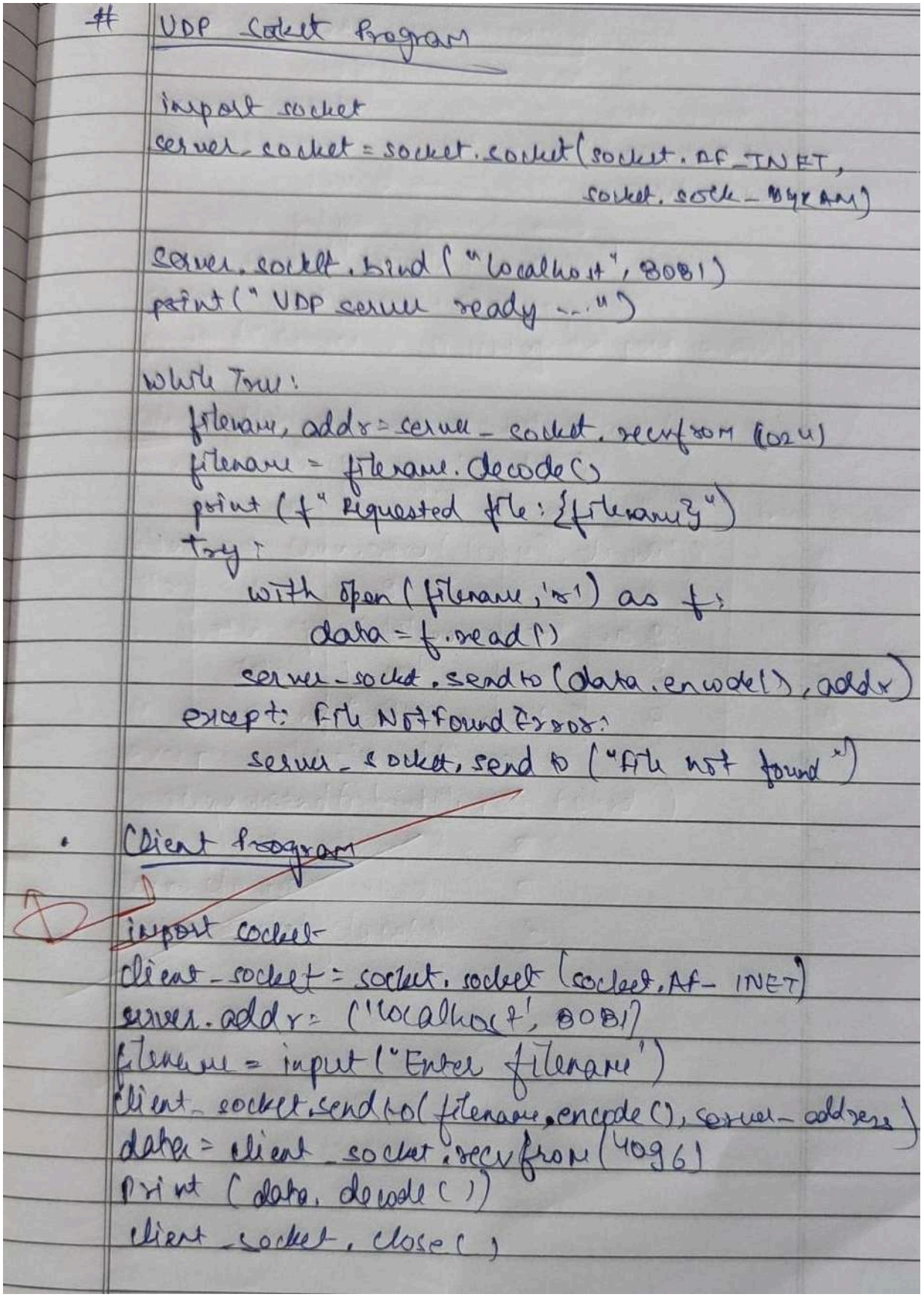
Output:





Program – 16:

Aim: 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.



UDP Socket Program

```
import socket
server_socket = socket.socket(socket.AF_INET,
                              socket.SOCK_DGRAM)

server.socket.bind("localhost", 8081)
print("UDP server ready ...")

while True:
    filename, addr = server_socket.recvfrom(1024)
    filename = filename.decode()
    print(f"Requested file: {filename}")
    try:
        with open(filename,'r') as f:
            data = f.read()
        server_socket.sendto(data.encode(), addr)
    except: File NotFoundError:
        server_socket.sendto("File not found")
```

- Client Program

```
import socket
client_socket = socket.socket(socket.AF_INET)
server_addr = ('localhost', 8081)
filename = input("Enter filename")
client_socket.sendto(filename.encode(), server_address)
data = client_socket.recvfrom(4096)
print(data.decode())
client_socket.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:

```
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")
```

```
The server is ready to receive

Sent contents ofServerUDP.py
```