VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

PRAJWAL R (1BM21CS135)

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
June-2023 to September-2023

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "COMPUTER NETWORKS" carried out by **PRAJWAL R (1BM21CS135),** who is bonafide student of **B.M.S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the academic semester June-2023 to September-2023. The Lab report has been approved as it satisfies the academic requirements in respect of a **COMPUTER NETWORKS (22CS4PCCON)** work prescribed for the said degree.

Swathi Sridharan	**Dr. Jyothi S Nayak**
Assistant Professor	Professor and Head
Department of CSE	Department of CSE
BMSCE, Bengaluru	BMSCE, Bengaluru

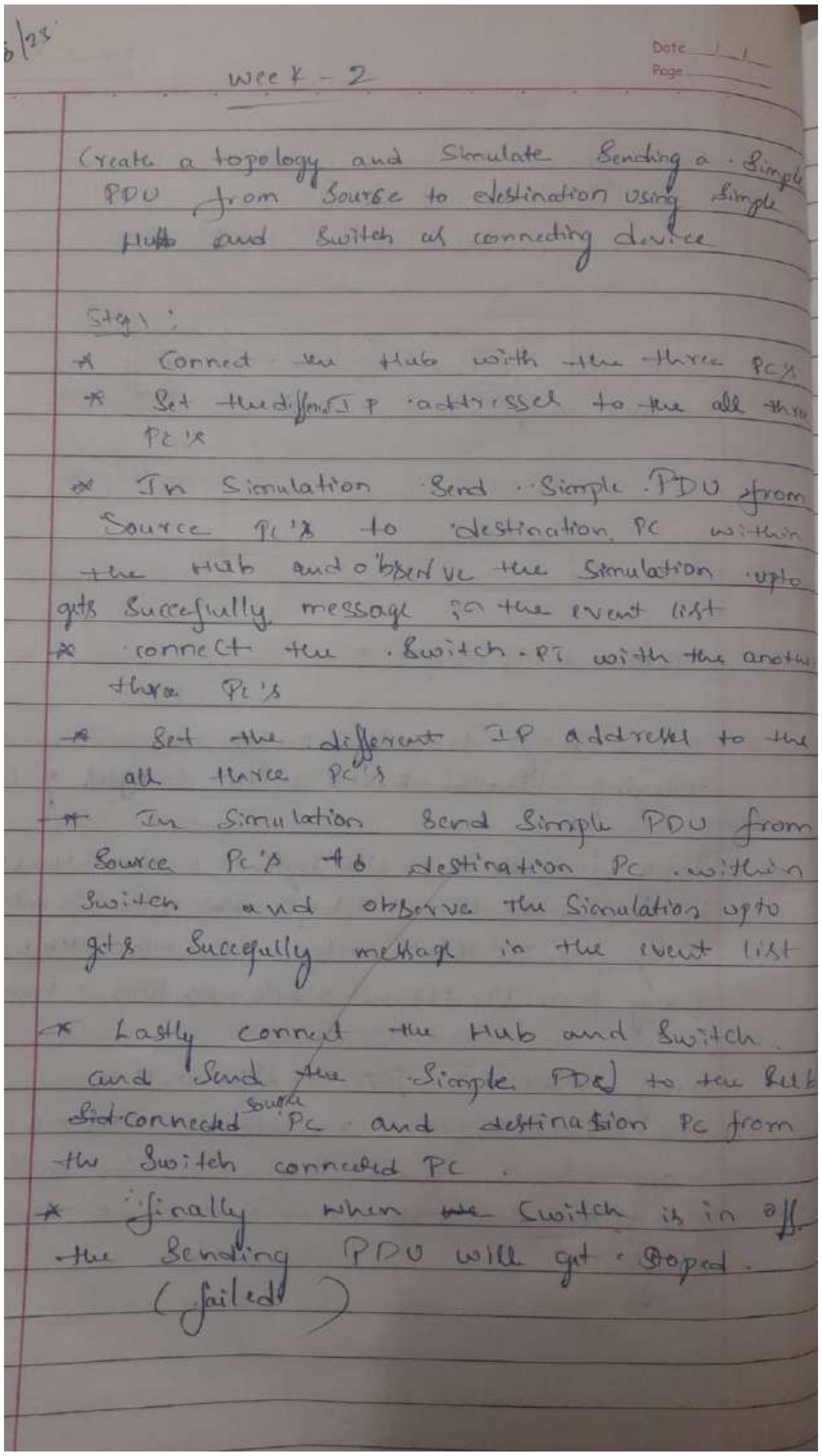
INDEX SHEET

Course Outcomes

CO1	Apply the fundamental concepts of communication in networking.
CO2	Analyze the various protocols, techniques in TCP/IP network architectur
CO3	Develop programs that demonstrate the functionalities of physical, Data Link, Network, Transport or Application layer

Experiment 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.



6/23

Date / /
Page

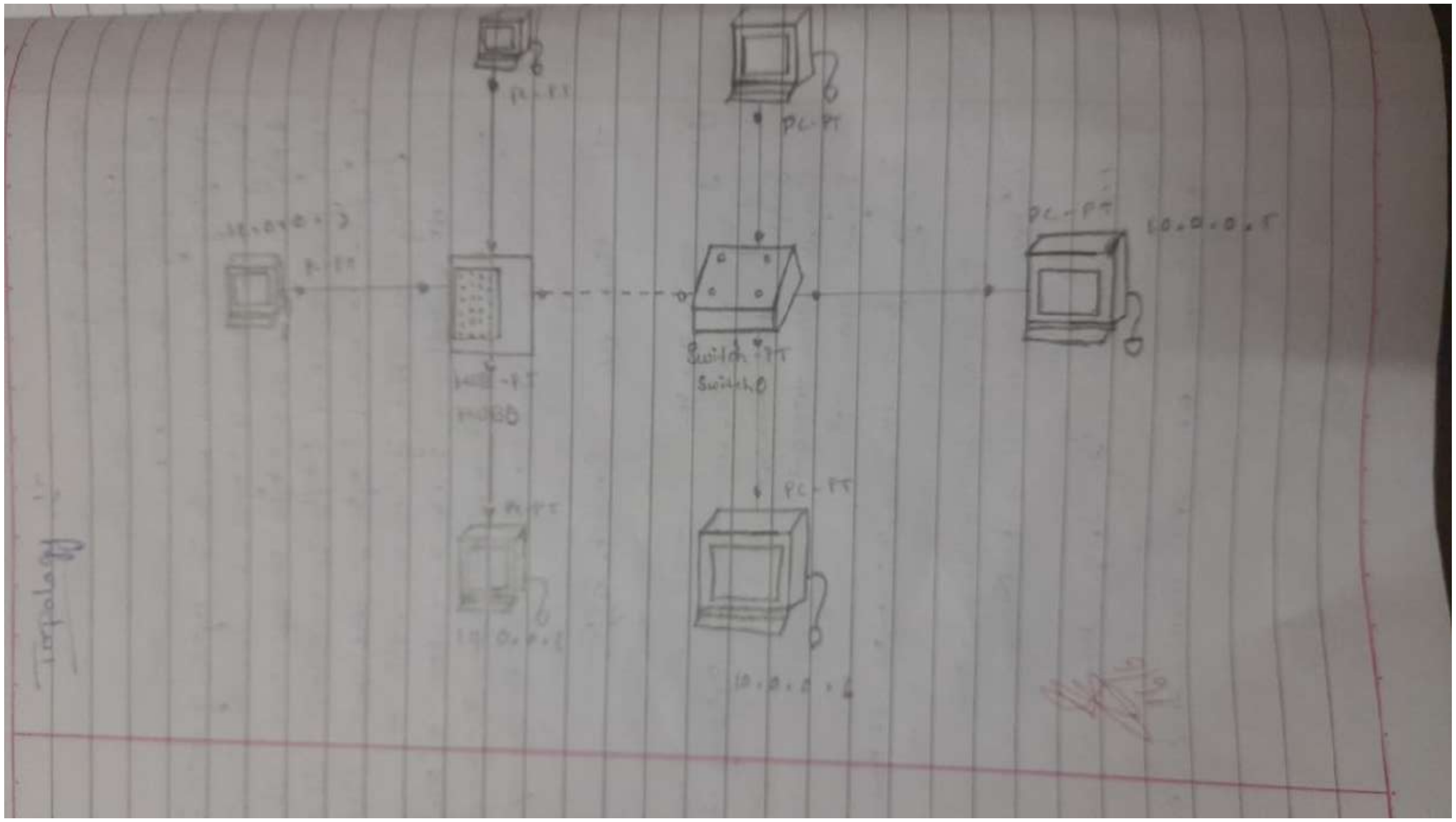
week - 2

Create a topology and Simulate Sending a Simple PDU from Source to destination using Simple Hub and Switch as connecting device

Step 1 :

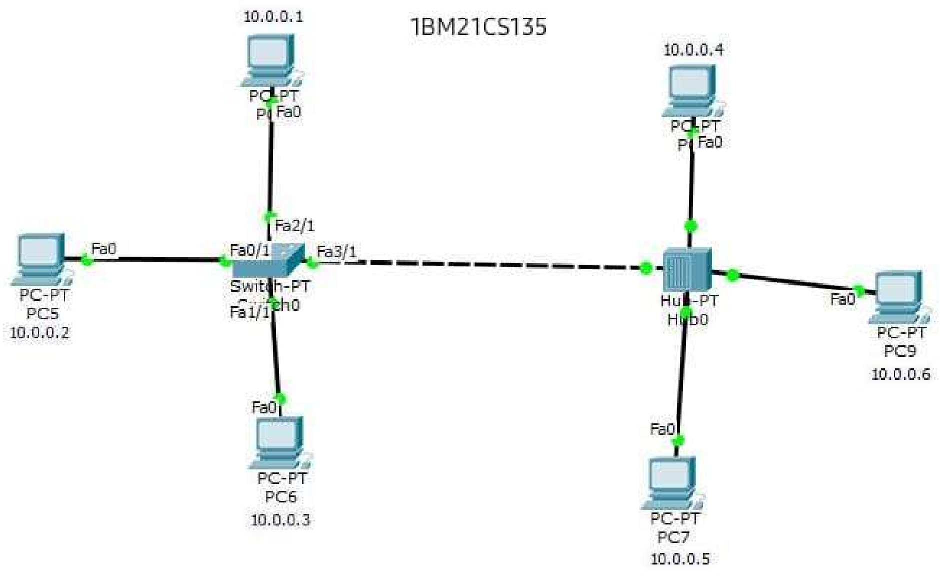
* Connect the Hub with the three PC's
* Set the different IP addrisses to the all three PC's
* In Simulation Send Simple PDU from Source Pc's to destination PC within the Hub and observe the Simulation upto gets Succefully message in the event list
* connect the Switch-PT with the another three Pc's
* Set the different IP addreses to the all three Pc's
* In Simulation Send Simple PDU from Source Pc's to destination Pc within Switch and observe the Simulation upto gets Succefully message in the event list
* Lastly connect the Hub and Switch and Send the Simple PDU to the left ~~Sid~~ connected source Pc and destination Pc from the Switch connected PC.
* finally when ~~we~~ Switch is in off the Sending PDU will get Droped. (failed)

Topology :-

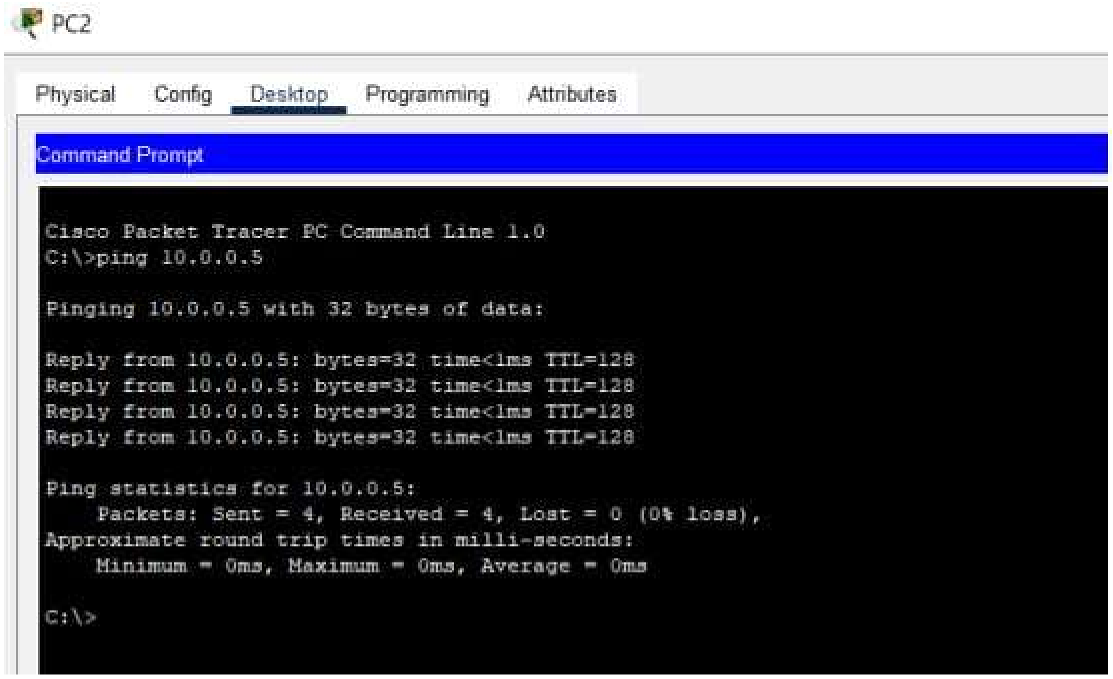


PC-PT
PC-PT
10.0.0.3
PC-PT
PC-PT
10.0.0.5
Switch-PT
Switch0
Hub-PT
Hub0
PC-PT
PC-PT
10.0.0.1
10.0.0.4

Topology:


10.0.0.1
1BM21CS135
10.0.0.4
PC-PT
Fa0
Fa0
Fa2/1
Fa0
Fa0/1
Fa3/1
PC-PT
PC5
10.0.0.2
Switch-PT
Switch0
Fa1/1
Hub-PT
Hub0
Fa0
PC-PT
PC9
10.0.0.6
Fa0
PC-PT
PC6
10.0.0.3
Fa0
PC-PT
PC7
10.0.0.5

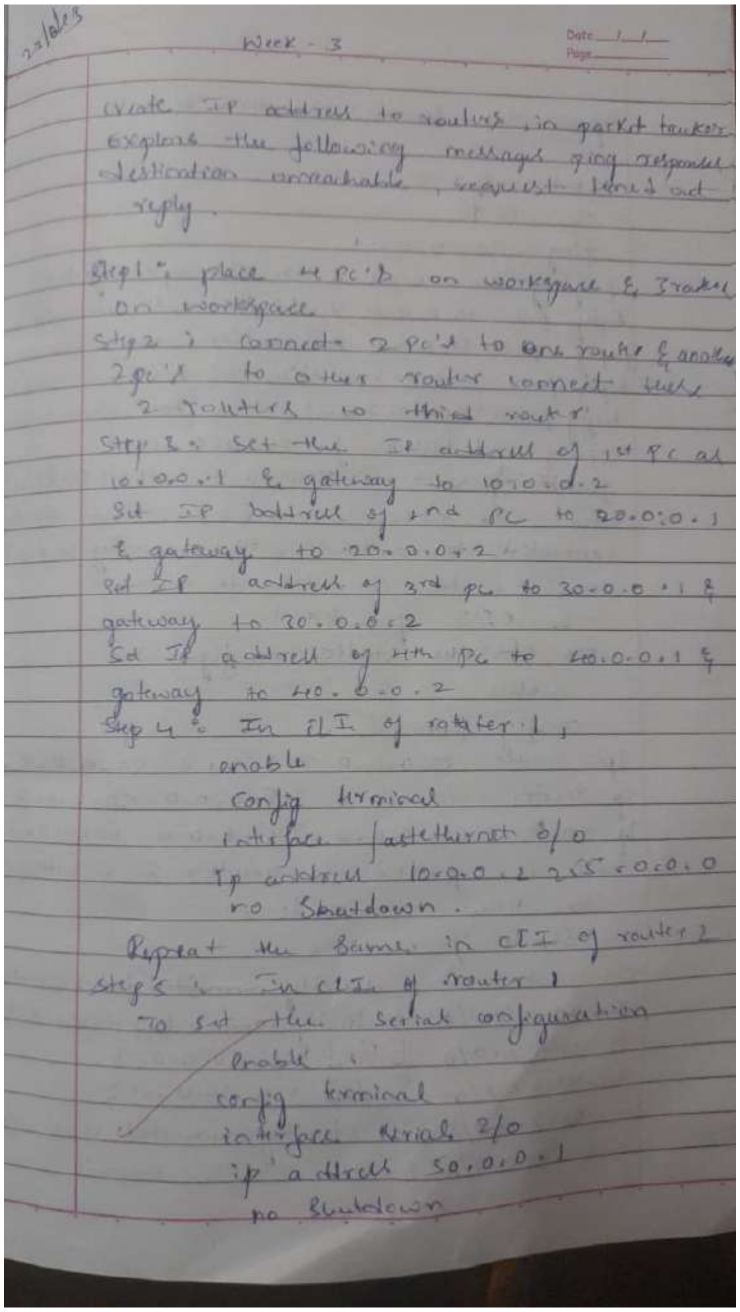
```
C:\>
```



```
Cisco Packet Tracer PC Command Line 1.0
C:\>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<1ms TTL=128
Reply from 10.0.0.5: bytes=32 time<1ms TTL=128
Reply from 10.0.0.5: bytes=32 time<1ms 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 = 0ms, Average = 0ms

C:\>
```

Experiment 2

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



22/02/23

Week - 3

Date / /
Page

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

Step 1 : place 4 PC's on workspace & 3 routers on workspace

Step 2 : connect 2 PC's to one router & another 2 pc's to other router connect these 2 routers to third router

Step 3 : Set the IP address of 1st PC as 10.0.0.1 & gateway to 10.0.0.2
Set IP address of 2nd PC to 20.0.0.1 & gateway to 20.0.0.2
Set IP address of 3rd pc to 30.0.0.1 & gateway to 30.0.0.2
Set IP address of 4th pc to 40.0.0.1 & gateway to 40.0.0.2

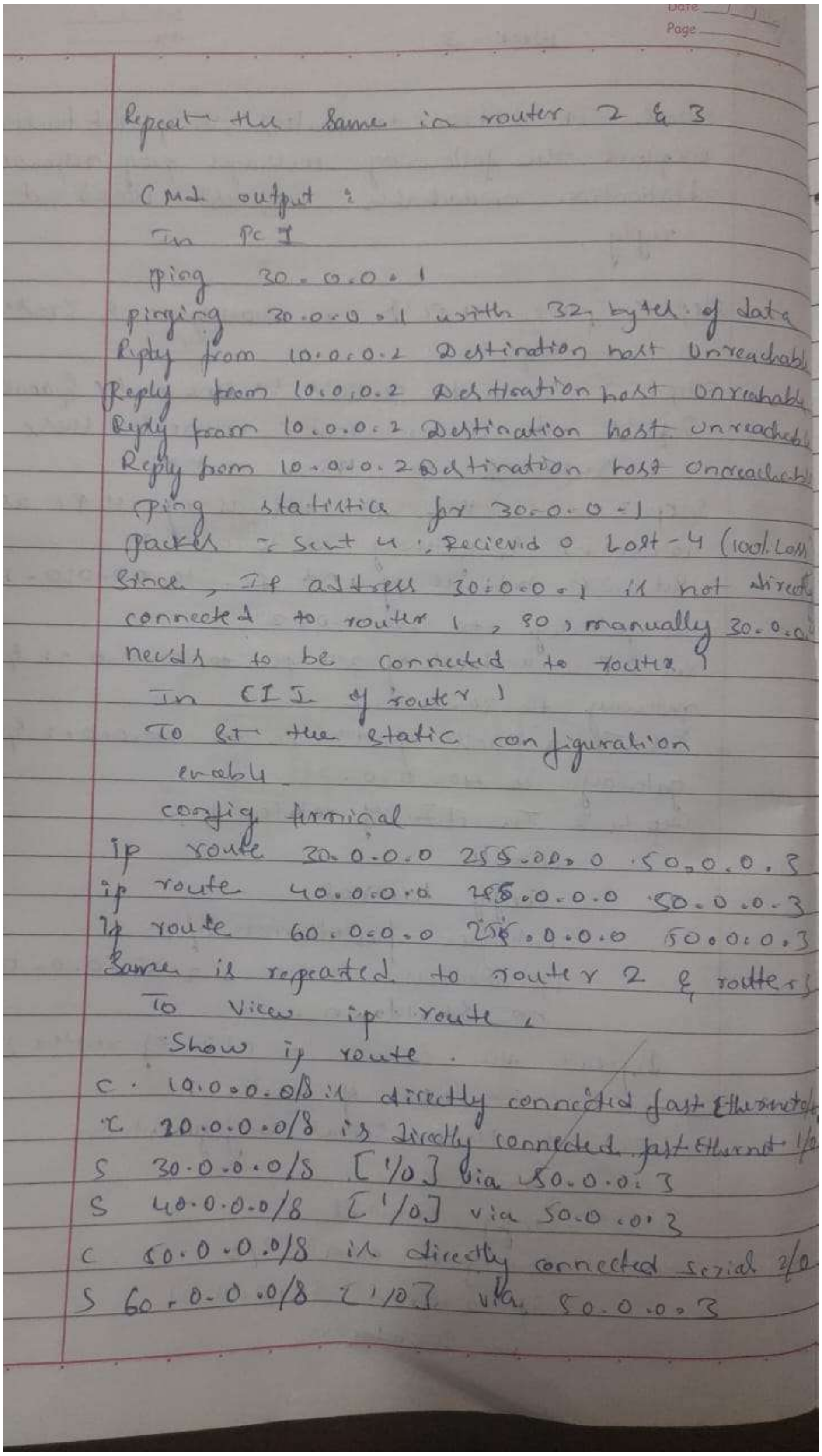
Step 4 : In CLI of router 1,

```
enable
config terminal
interface fastethernet 0/0
ip address 10.0.0.2 255.0.0.0
no shutdown.
```

Repeat the same in CLI of router 2

Step 5 : In CLI of router 1
To set the serial configuration

```
enable
config terminal
interface serial 2/0
ip address 50.0.0.1
no shutdown
```

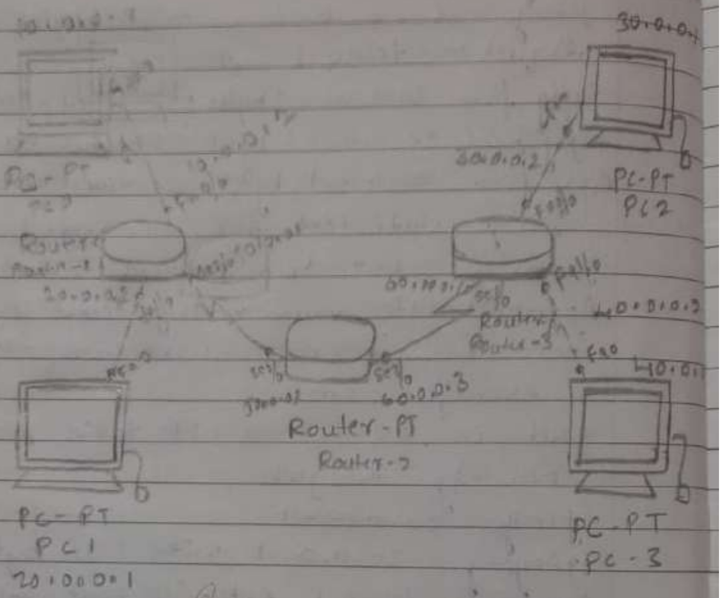


Date
Page

Repeat the Same in router 2 & 3

CMd output :
In PC 1
ping 30.0.0.1
pinging 30.0.0.1 with 32 bytes of data
Reply from 10.0.0.2 Destination host Unreachable
Reply from 10.0.0.2 Destination host Unreachable
Reply from 10.0.0.2 Destination host unreachable
Reply from 10.0.0.2 Destination host Unreachable
Ping statistics for 30.0.0.1,
Packets : Sent 4 ; Recieved 0 Lost - 4 (100% Loss
Since, IP address 30.0.0.1 is not directly
connected to router 1, so, manually 30.0.0
needs to be connected to router 1
In CLI of router 1
To set the static configuration
enable
config terminal
ip route 30.0.0.0 255.0.0.0 50.0.0.3
ip route 40.0.0.0 255.0.0.0 50.0.0.3
ip route 60.0.0.0 255.0.0.0 50.0.0.3
Same is repeated to router 2 & router 3
To View ip route,
Show ip route.
C. 10.0.0.0/8 is directly connected fast Ethernet 0/
C 20.0.0.0/8 is directly connected fast Ethernet 1/0
S 30.0.0.0/8 [1/0] via 50.0.0.3
S 40.0.0.0/8 [1/0] via 50.0.0.3
C 50.0.0.0/8 is directly connected serial 2/0
S 60.0.0.0/8 [1/0] via 50.0.0.3

Topology



[illegible] 26/6/25

Date __/__/__
Page ______

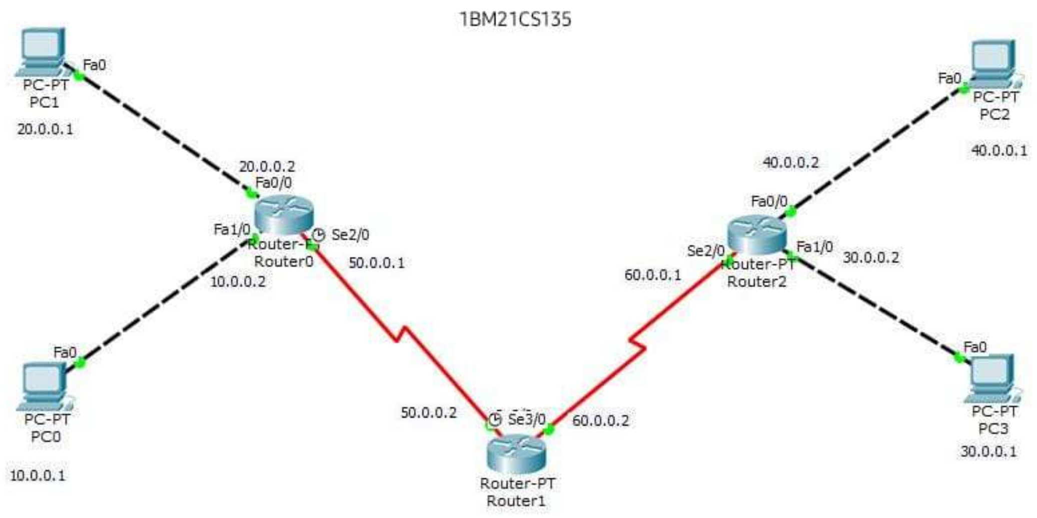
In PC 1

```
ping 30.0.0.1
Ping 30.0.0.1 with 32 bytes of data.
Request timed out.
Reply from 30.0.0.1: bytes=32 time=5ms TTL=125
Reply from 30.0.0.1: bytes=32 time=8ms TTL=125
Reply from 30.0.0.1: bytes=32 time=10ms TTL=125
 Ping statistics for 30.0.0.1
packets: sent 4, Recieved 3 lost=1 (25% loss
Approximate round trip times in millisec
 Minimum = 2ms , Maximum = 10ms,
  Average = 5ms
```

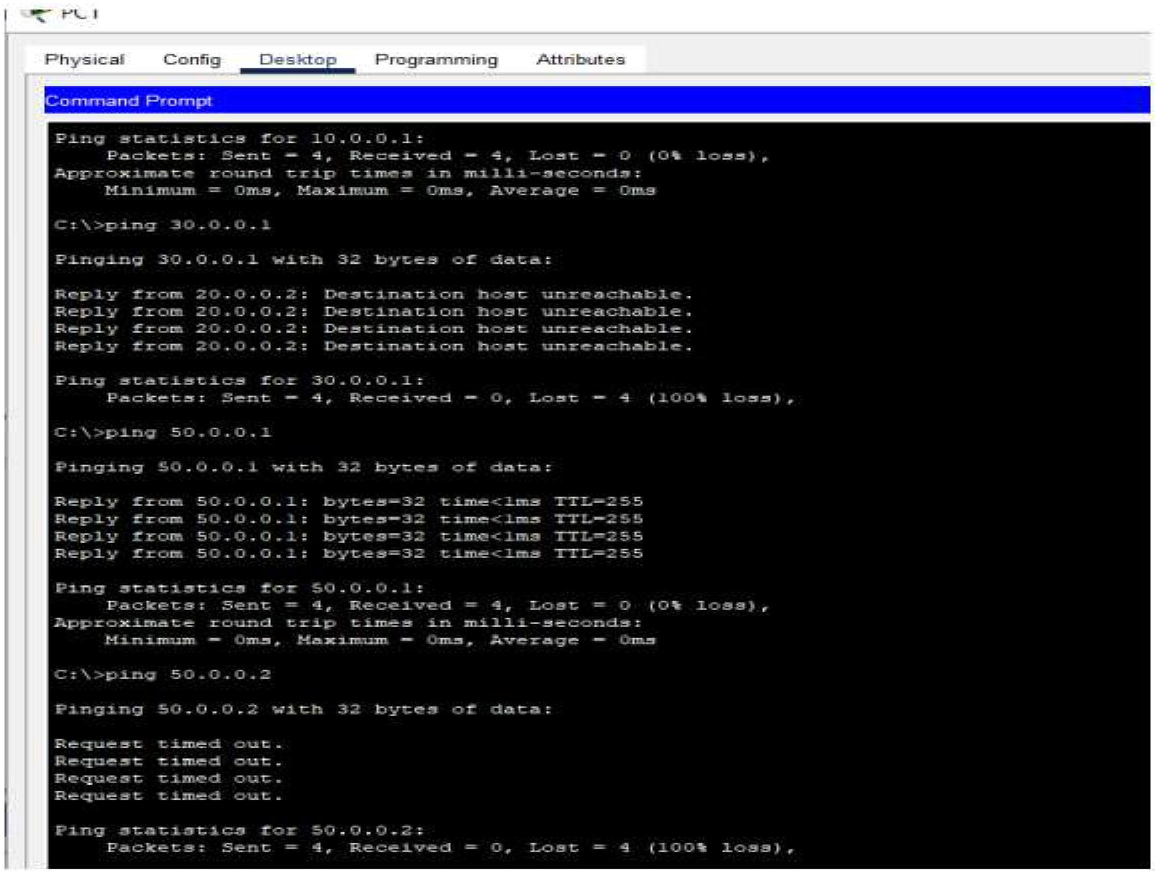
first is lost since it takes time to
 identify the path.

```
ping 30.0.0.1
pinging 30.0.0.1 with 32 bytes of data
Reply from 30.0.0.1 bytes=32 time=3ms TTL=125
Reply from 30.0.0.1 bytes=32 time=11ms TTL=125
Reply from 30.0.0.1 bytes=32 time=4ms TTL=125
Reply from 30.0.0.1 bytes=32 time=6ms TTL=125
 Ping statistics for 30.0.0.1 :
   packet sent =4 , received =4, Lost =0 (0% loss
Approximate round trip time in milli sec
 Minimum = 3ms , maximum =11ms , Average =6m
```

Topology:



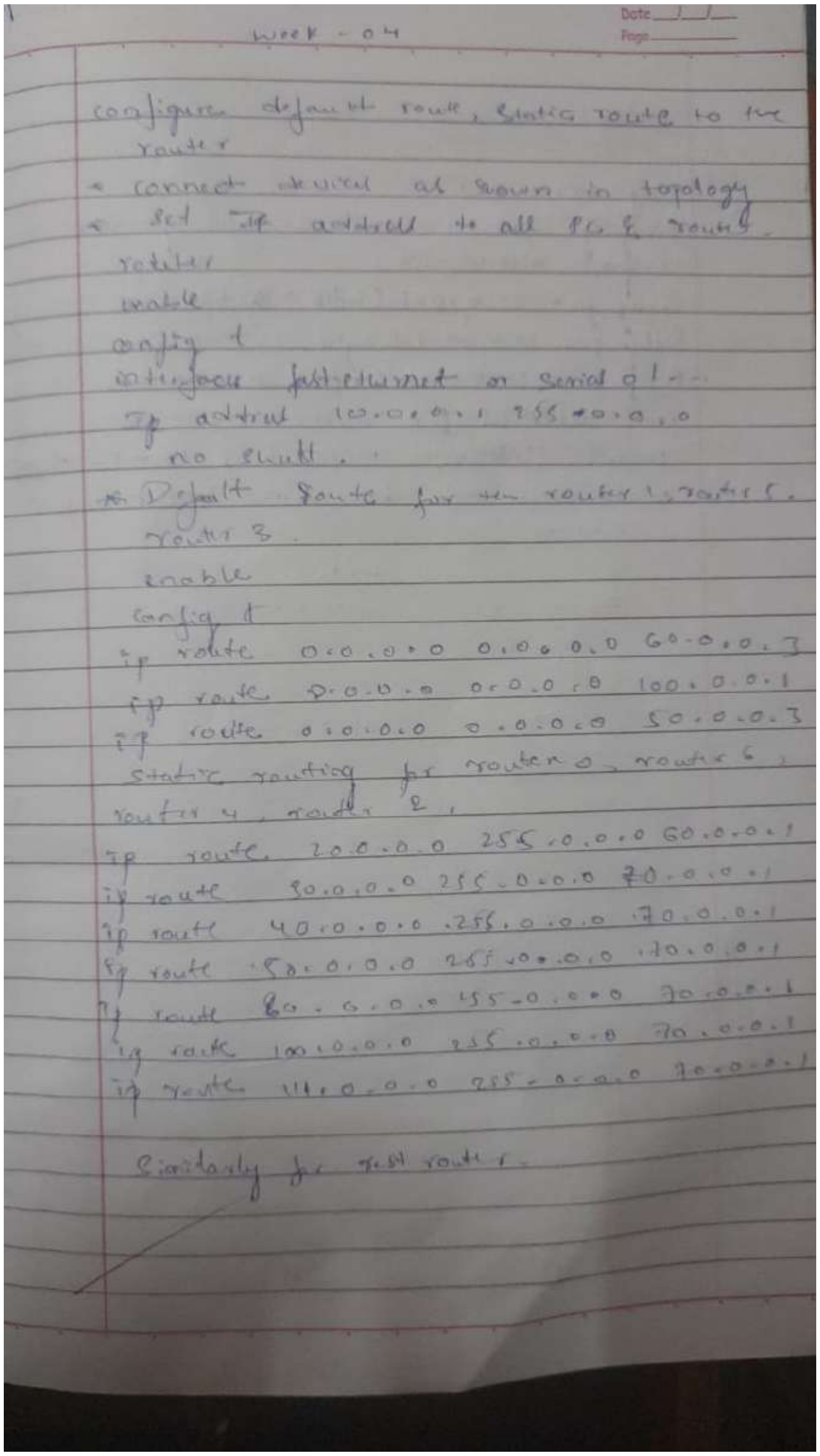
Output:



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

C:\>ping 30.0.0.1

Pinging 30.0.0.1 with 32 bytes of data:

Reply from 20.0.0.2: Destination host unreachable.
Reply from 20.0.0.2: Destination host unreachable.
Reply from 20.0.0.2: Destination host unreachable.
Reply from 20.0.0.2: Destination host unreachable.

Ping statistics for 30.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>ping 50.0.0.1

Pinging 50.0.0.1 with 32 bytes of data:

Reply from 50.0.0.1: bytes=32 time<1ms TTL=255
Reply from 50.0.0.1: bytes=32 time<1ms TTL=255
Reply from 50.0.0.1: bytes=32 time<1ms TTL=255
Reply from 50.0.0.1: bytes=32 time<1ms TTL=255

Ping statistics for 50.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>ping 50.0.0.2

Pinging 50.0.0.2 with 32 bytes of data:

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

Ping statistics for 50.0.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Experiment 3

Configure default route, static route to the Router



Date __/__/__
Page ____

week - 04

configure default route, static route to the router

- connect devices as shown in topology
- set IP address to all PC & routers.

```
router
enable
config t
interface fastethernet or serial 0/1 ..
 ip address 10.0.0.1 255.0.0.0
  no shutd.
```

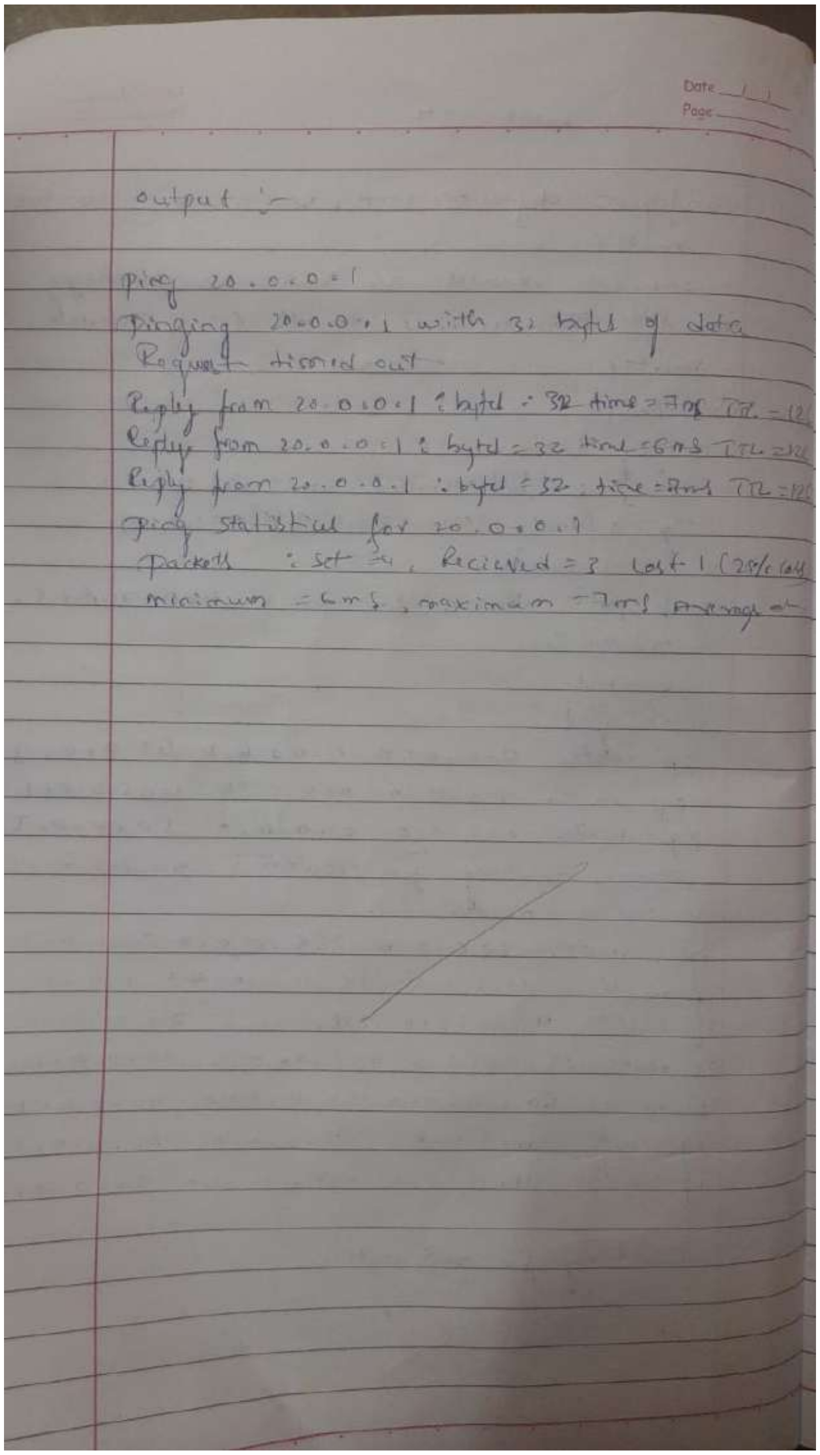
- Default route for the router 1, router 5, router 3.

```
enable
config t
ip route 0.0.0.0 0.0.0.0 60.0.0.3
ip route 0.0.0.0 0.0.0.0 100.0.0.1
ip route 0.0.0.0 0.0.0.0 50.0.0.3
```

static routing for router 0, router 6, router 4, router 2,

```
ip route 20.0.0.0 255.0.0.0 60.0.0.1
ip route 30.0.0.0 255.0.0.0 70.0.0.1
ip route 40.0.0.0 .255.0.0.0 .70.0.0.1
ip route .50.0.0.0 255.0.0.0 .70.0.0.1
ip route 80.0.0.0 255.0.0.0 70.0.0.1
ip route 100.0.0.0 255.0.0.0 70.0.0.1
ip route 11.0.0.0 255.0.0.0 70.0.0.1
```

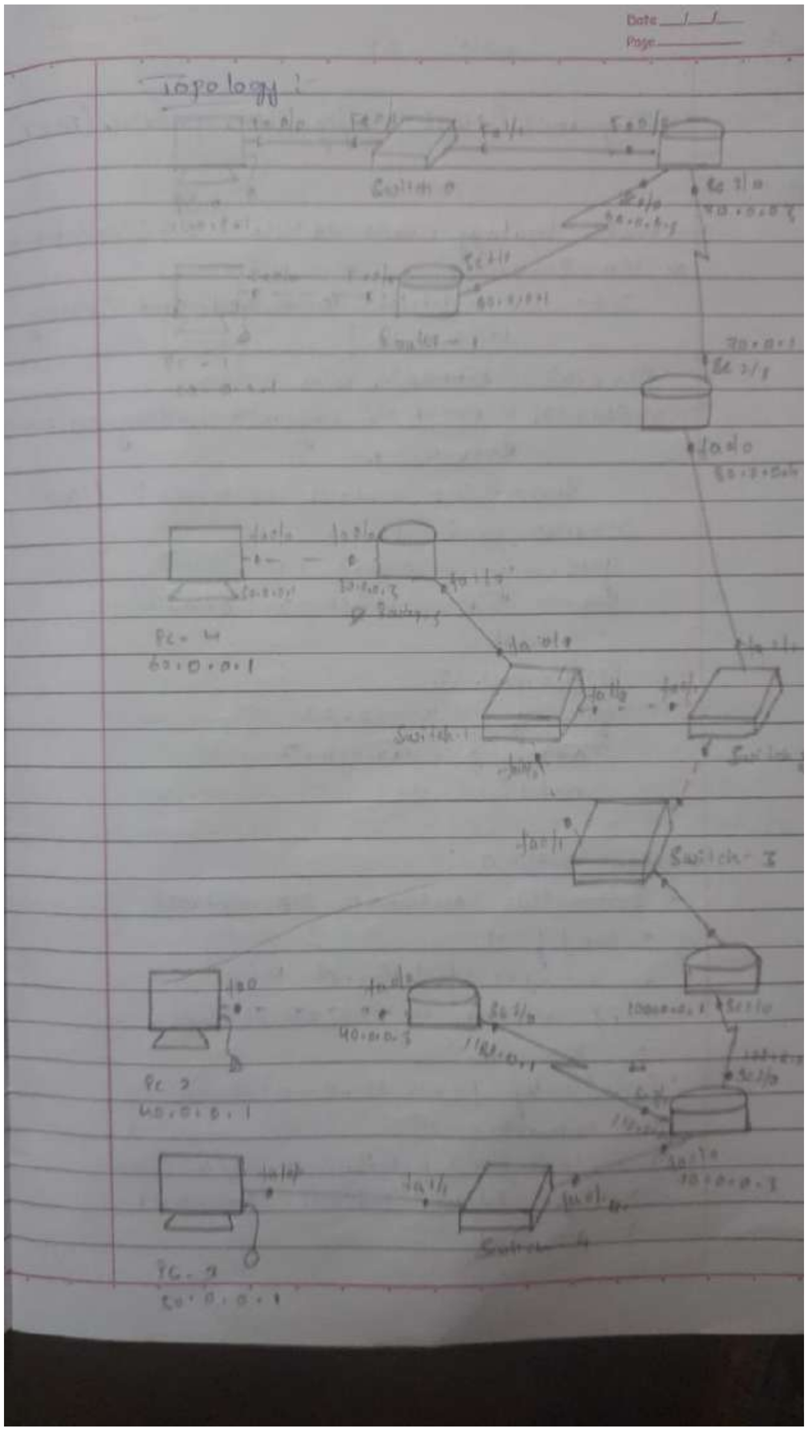
Similarly for rest router.



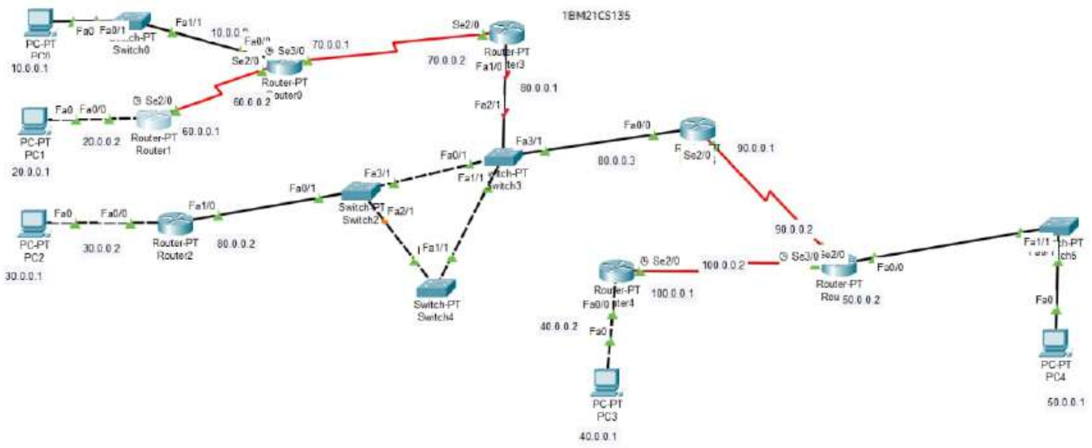
Date __/__/__
Page ____

output :-

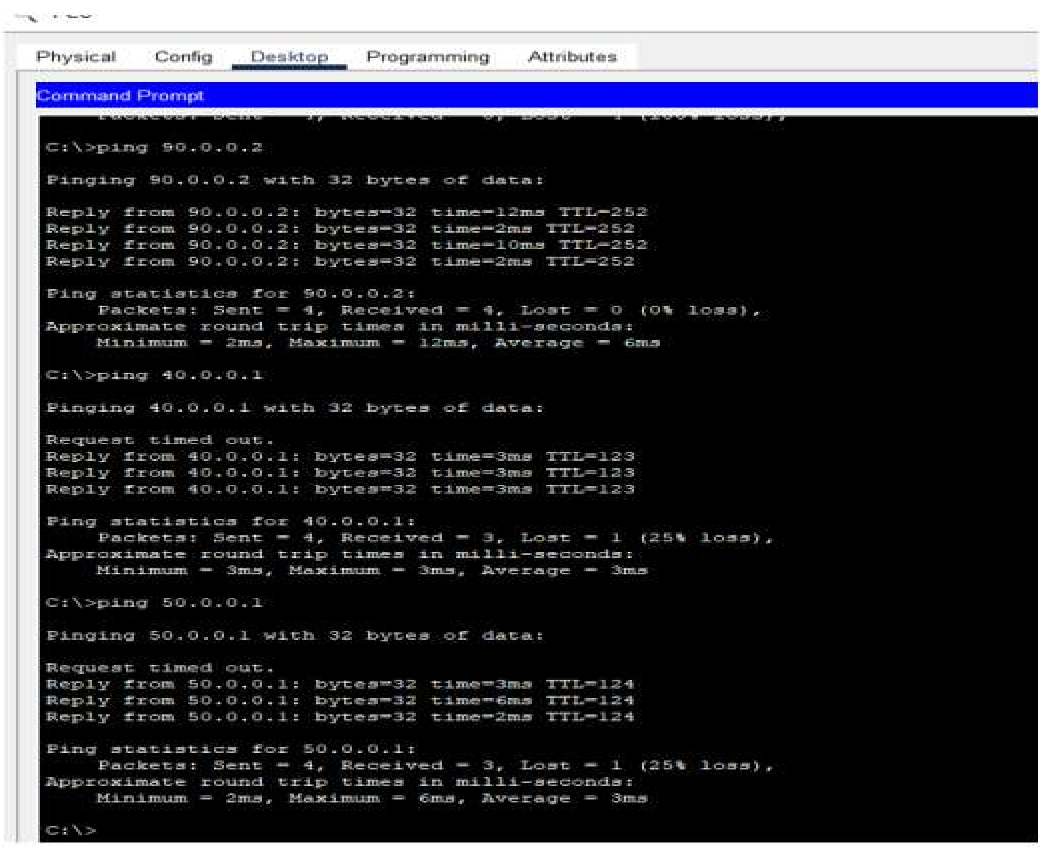
```
ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data
Request timed out
Reply from 20.0.0.1 : bytes = 32 time = 7ms TTL = 12
Reply from 20.0.0.1 : bytes = 32 time = 6ms TTL = 12
Reply from 20.0.0.1 : bytes = 32 time = 7ms TTL = 12
Ping statistics for 20.0.0.1
Packets : set = 4, Recieved = 3 Lost 1 (25% loss
minimum = 6ms, maximum = 7ms Average =
```


Date / /
Page
Topology :
Switch 0
Switch 1
Switch 2
Switch 3
Switch 4
PC 4
60.0.0.1
PC 2
40.0.0.1
PC 3
50.0.0.1
40.0.0.3
70.0.0.1
70.0.0.3

Topology:



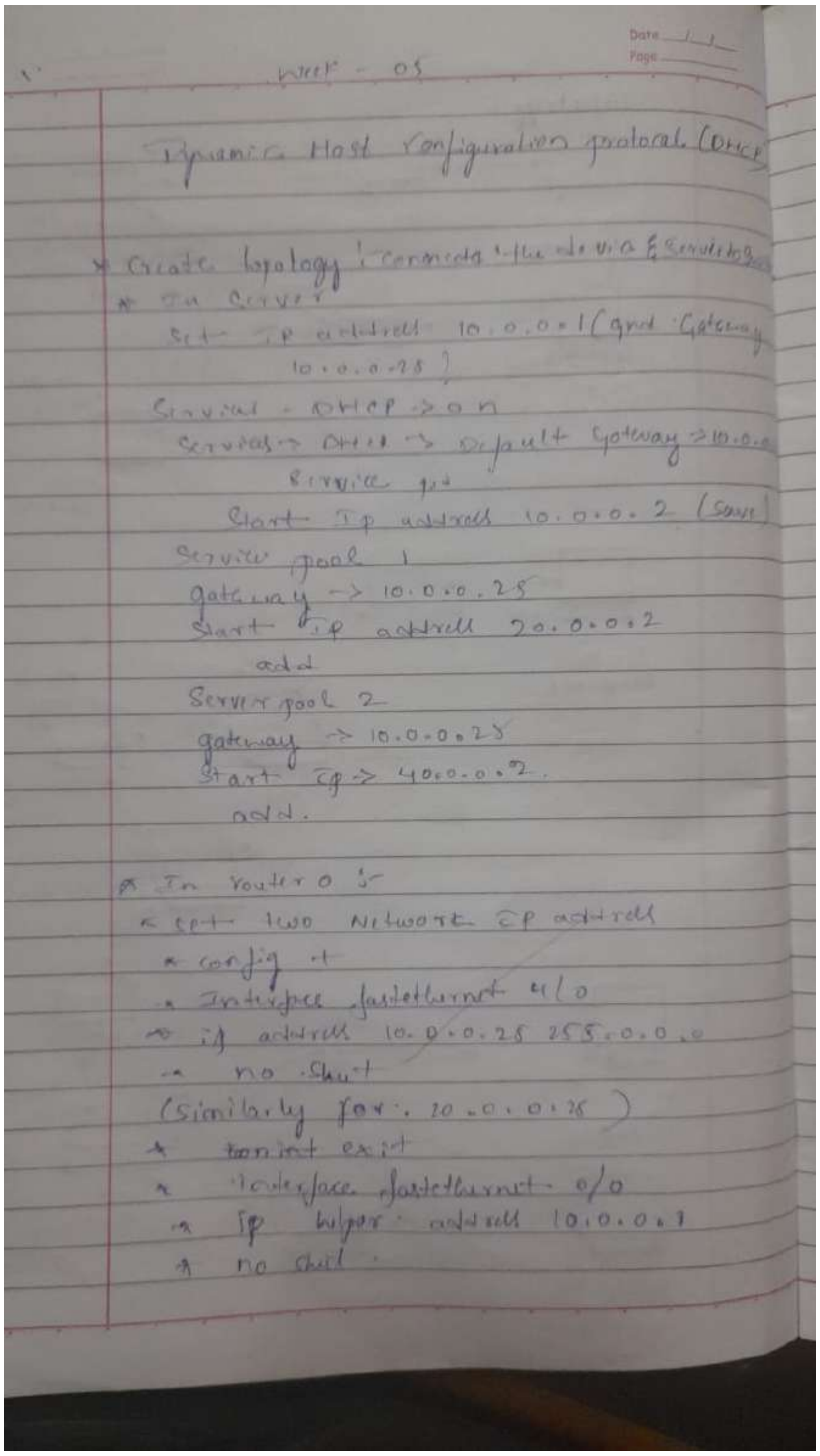
Output:



```
C:\>ping 90.0.0.2

Pinging 90.0.0.2 with 32 bytes of data:

Reply from 90.0.0.2: bytes=32 time=12ms TTL=252
Reply from 90.0.0.2: bytes=32 time=2ms TTL=252
Reply from 90.0.0.2: bytes=32 time=10ms TTL=252
Reply from 90.0.0.2: bytes=32 time=2ms TTL=252

Ping statistics for 90.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 12ms, Average = 6ms

C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=3ms TTL=123
Reply from 40.0.0.1: bytes=32 time=3ms TTL=123
Reply from 40.0.0.1: bytes=32 time=3ms TTL=123

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

C:\>ping 50.0.0.1

Pinging 50.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 50.0.0.1: bytes=32 time=3ms TTL=124
Reply from 50.0.0.1: bytes=32 time=6ms TTL=124
Reply from 50.0.0.1: bytes=32 time=2ms TTL=124

Ping statistics for 50.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 6ms, Average = 3ms

C:\>
```

Experiment 4

Configure DHCP within a LAN and outside LAN



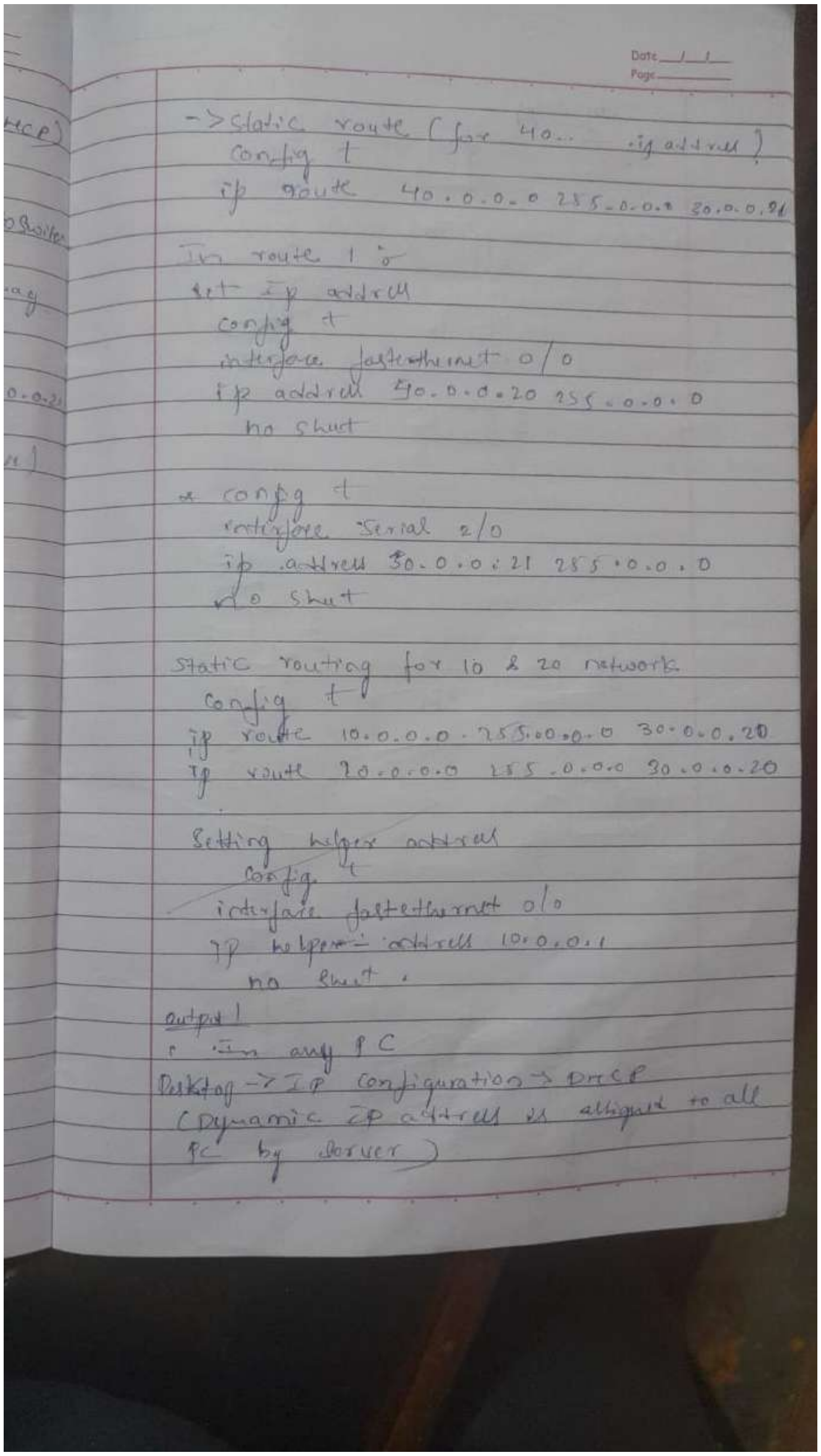
Date / /
Page

Week - 05

Dynamic Host Configuration protocol (DHCP)

* Create topology : connect the devices & server
* In Server
  Set IP address 10.0.0.1 (and Gateway 10.0.0.25)
  Services - DHCP -> on
  Services -> DHCP -> Default gateway -> 10.0.0
  Service pool
  Start Ip address 10.0.0.2 (Save)
  Service pool 1
  gateway -> 10.0.0.25
  Start ip address 20.0.0.2
  add
  Server pool 2
  gateway -> 10.0.0.25
  Start ip -> 40.0.0.2.
  add.

* In router 0 :-
  * set two Network ip address
  * config t
  * Interface fastethernet 4/0
  * ip address 10.0.0.25 255.0.0.0
  * no shut
  (Similarly for : 20.0.0.25)
  * ~~[illegible]~~ exit
  * Interface fastethernet 0/0
  * ip helper address 10.0.0.1
  * no shut.



-> Static route (for 40.. ip address)

```
config t
ip route 40.0.0.0 255.0.0.0 30.0.0.21
```

In route 1 :

set ip address

```
config t
interface fastethernet 0/0
ip address 40.0.0.20 255.0.0.0
no shut
```

```
config t
interface Serial 2/0
ip address 30.0.0.21 255.0.0.0
no shut
```

Static routing for 10 & 20 networks

```
config t
ip route 10.0.0.0 255.0.0.0 30.0.0.20
ip route 20.0.0.0 255.0.0.0 30.0.0.20
```

Setting helper address

```
config t
interface fastethernet 0/0
ip helper-address 10.0.0.1
no shut.
```

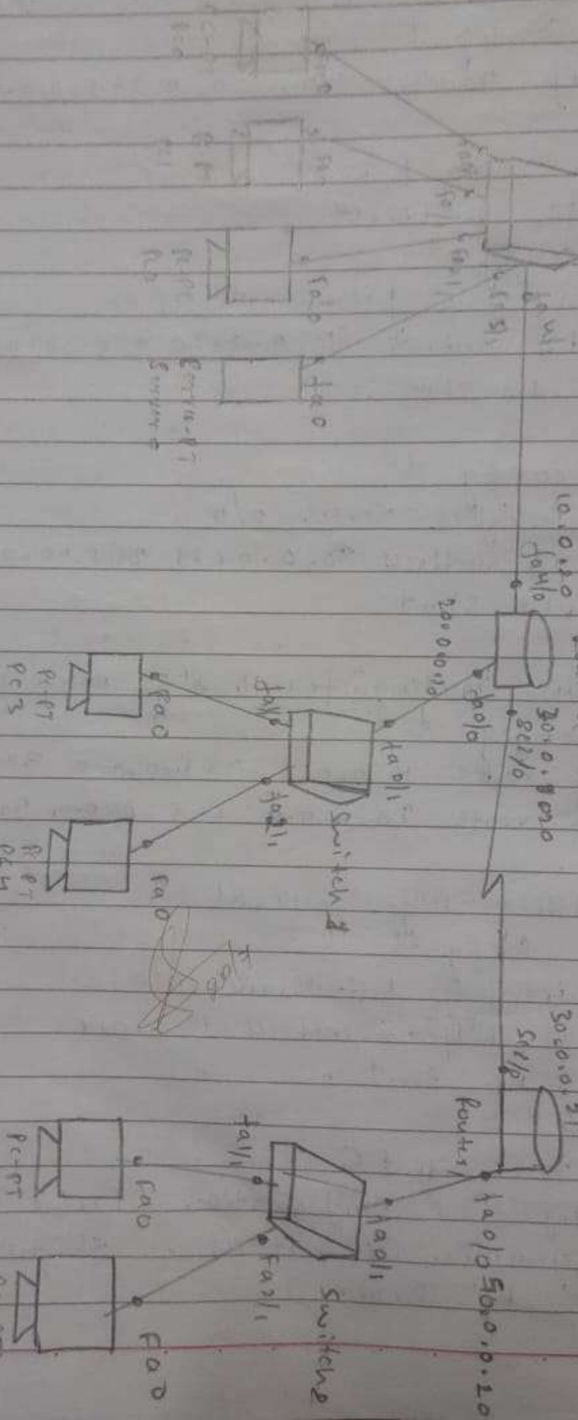
Output:

In any PC

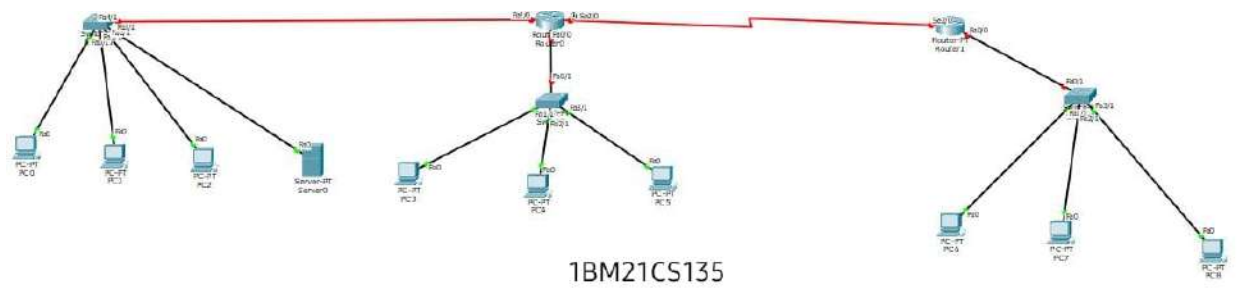
Desktop -> IP configuration -> DHCP

(Dynamic IP address is assigned to all PC by server)

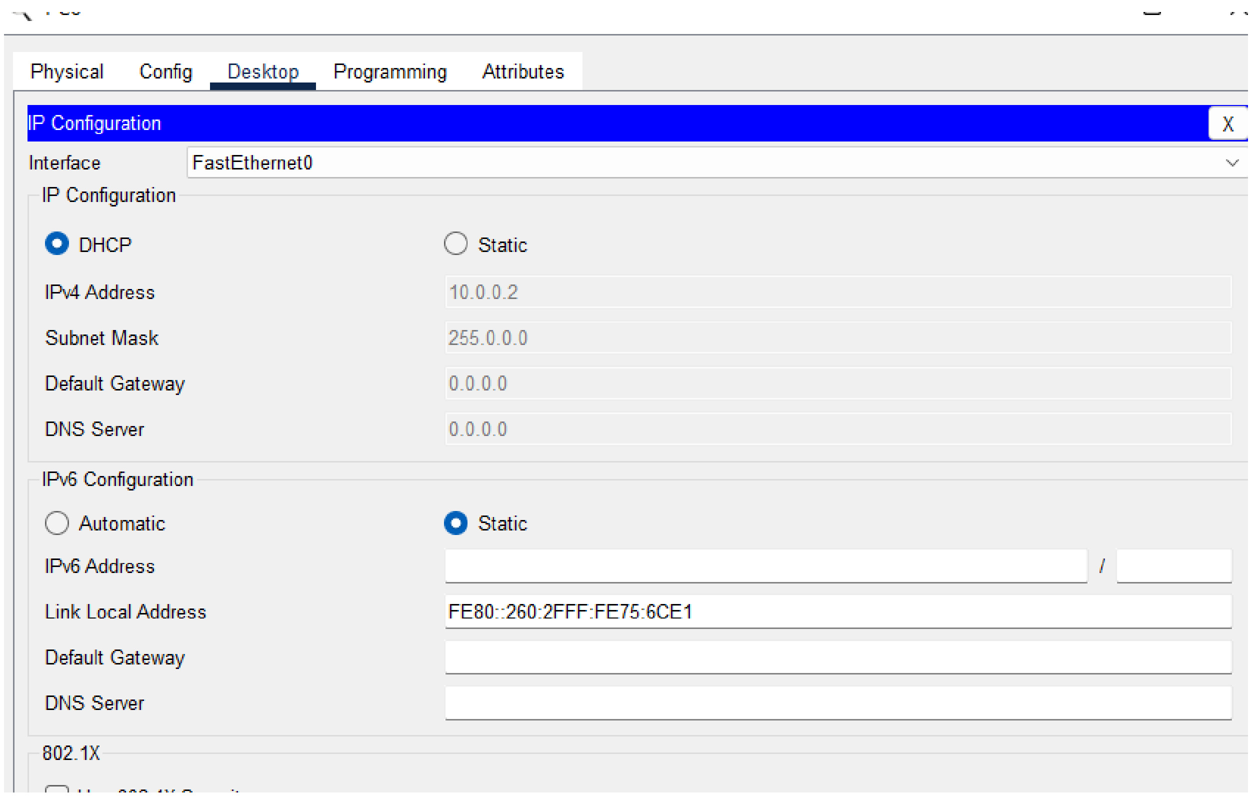
Topology :


Switch 0
Server-PT
Server0
10.0.0.20
fa0/0
Router0
20.0.0.10
fa0/1
30.0.0.20
Se2/0
Switch 1
Fa0/1
PC-PT
PC3
PC-PT
PC4
Fa0
30.0.0.21
Se2/0
Router1
fa0/0
40.0.0.20
Fa0/1
Switch2
PC-PT
PC5
PC-PT
PC-6
Fa0

Topology:



Output:



Physical | Config | Desktop | Programming | Attributes

IP Configuration X

Interface FastEthernet0

IP Configuration

DHCP [✓] Static

IPv4 Address 10.0.0.2

Subnet Mask 255.0.0.0

Default Gateway 0.0.0.0

DNS Server 0.0.0.0

IPv6 Configuration

Automatic Static [✓]

IPv6 Address /

Link Local Address FE80::260:2FFF:FE75:6CE1

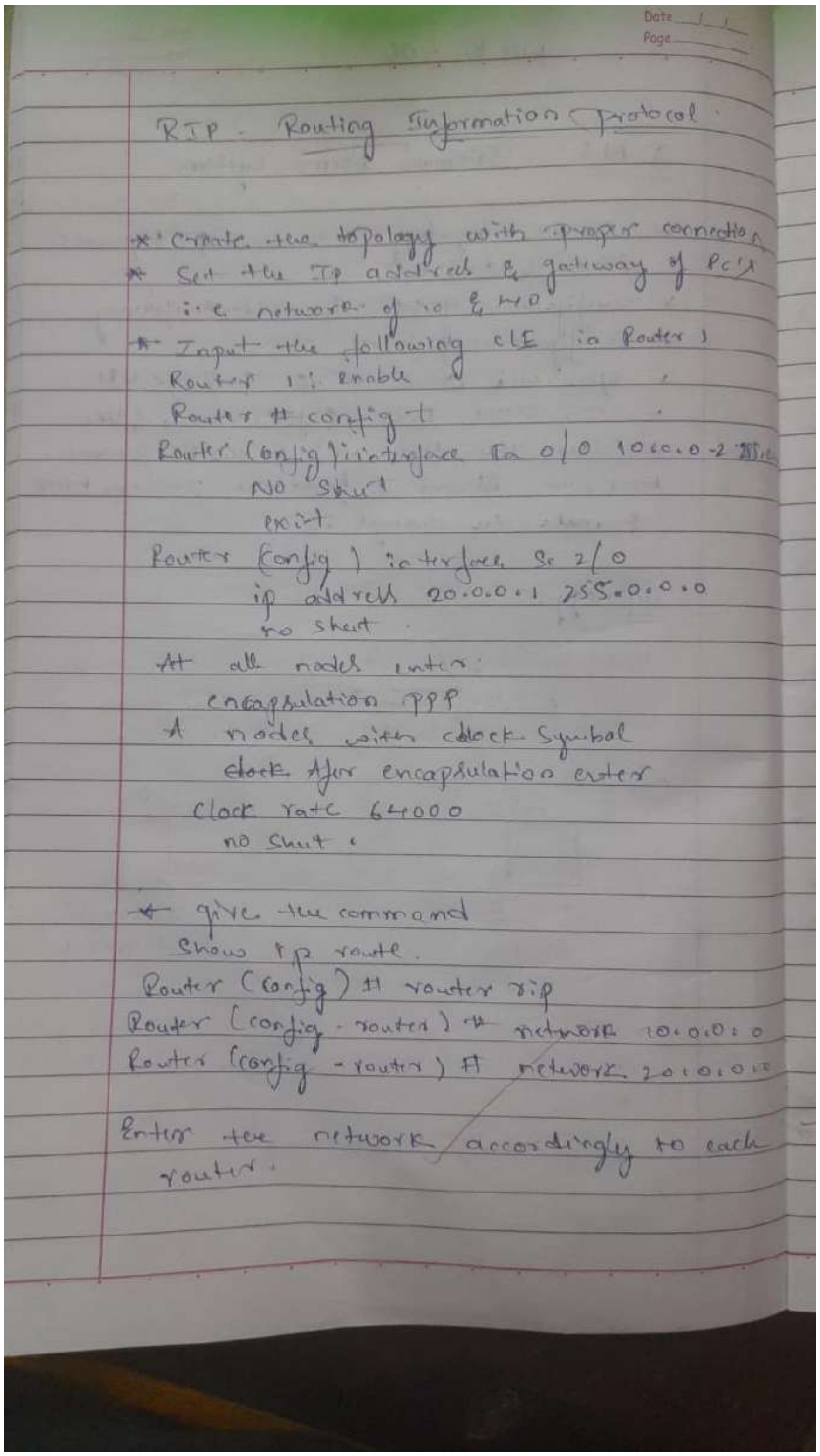
Default Gateway

DNS Server

802.1X

Experiment 5

Configure RIP routing Protocol in Routers



Date __/__/__
Page ____

RIP - Routing Information Protocol.

* Create the topology with proper connection
* Set the IP address & gateway of PC's i.e network of 10 & 40
* Input the following CLI in Router 1

```
Router 1: enable
Router # config t
Router (config)# interface Fa 0/0 10.0.0.2 255.0
        NO Shut
        exit
Router (config ) interface Se 2/0
        ip address 20.0.0.1 255.0.0.0
        no shut
```

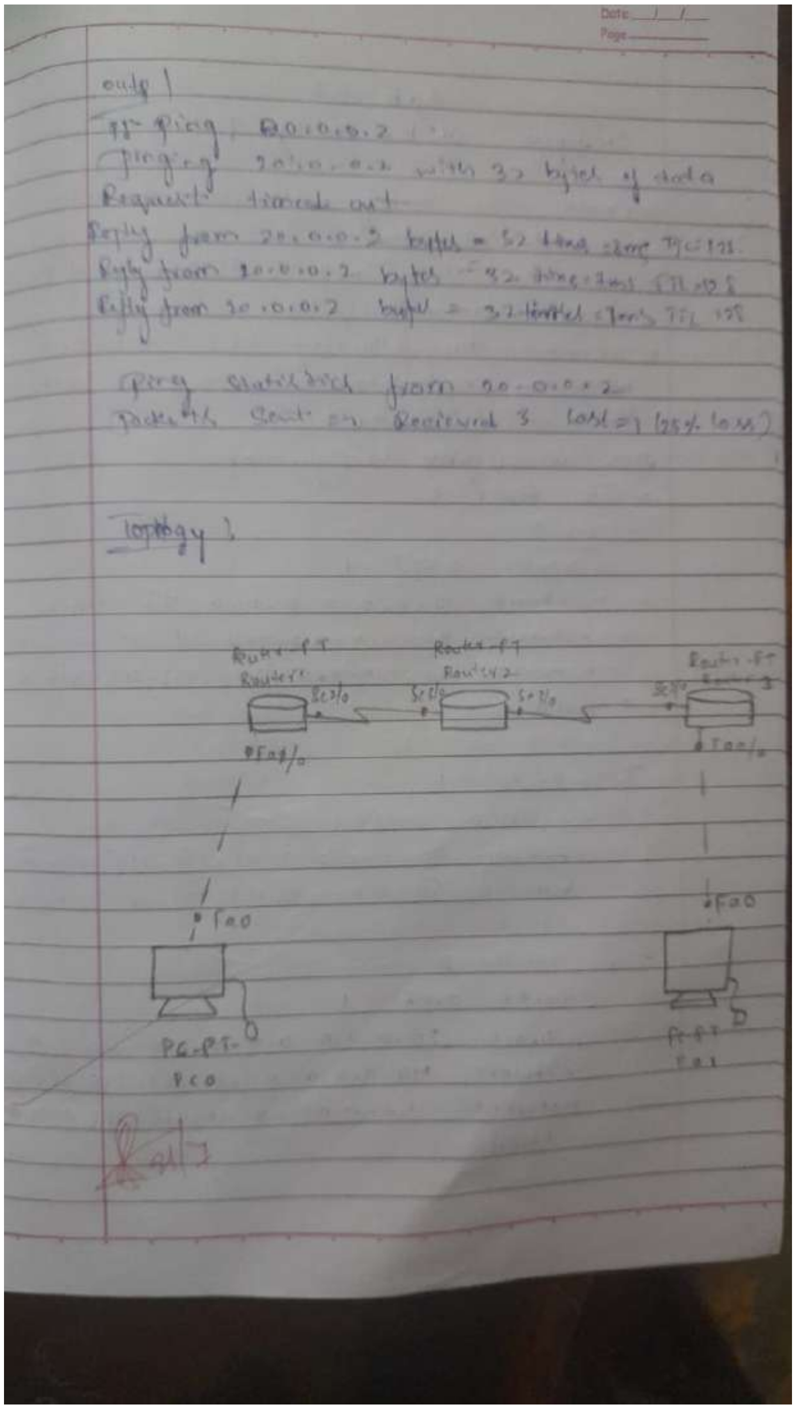
At all nodes enter:
encapsulation PPP

A nodes with clock symbol clock After encapsulation enter
clock rate 64000
no shut.

* give the command
Show ip route.

```
Router (config)# router rip
Router (config - router) # network 10.0.0.0
Router (config - router) # network 20.0.0.0
```

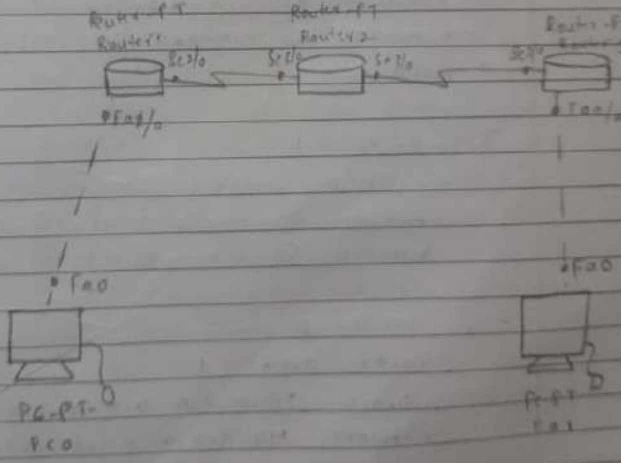
Enter the network accordingly to each router.



Date __/__/__
Page _____

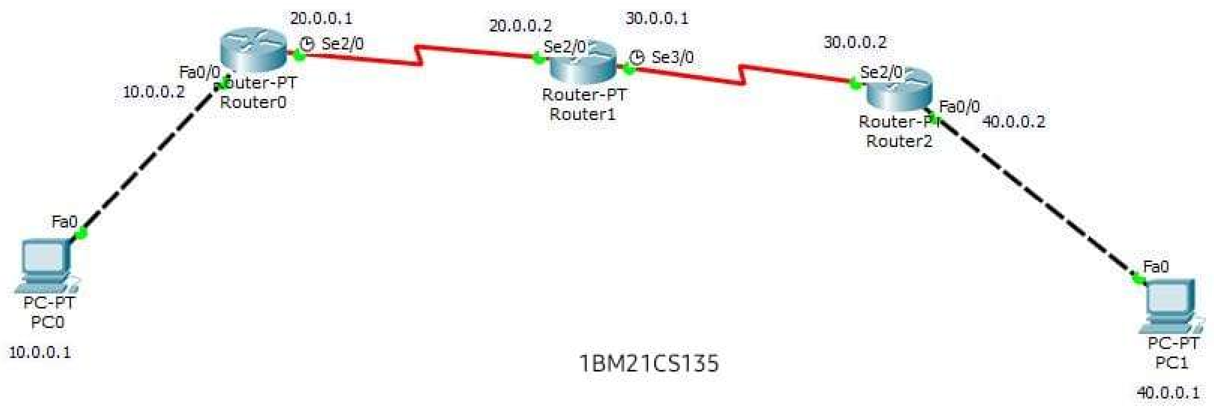
```
outp !
PC> ping 20.0.0.2
Pinging 20.0.0.2 with 32 bytes of data
Request timed out
Reply from 20.0.0.2 bytes = 32 time =2ms TTL=125
Reply from 20.0.0.2 bytes = 32 time=2ms TTL=125
Reply from 20.0.0.2 bytes = 32 time<1ms TTL 125

Ping statistics from 20.0.0.2
Packets Sent =4 Received 3 Lost =1 (25% loss)
```

Topology :

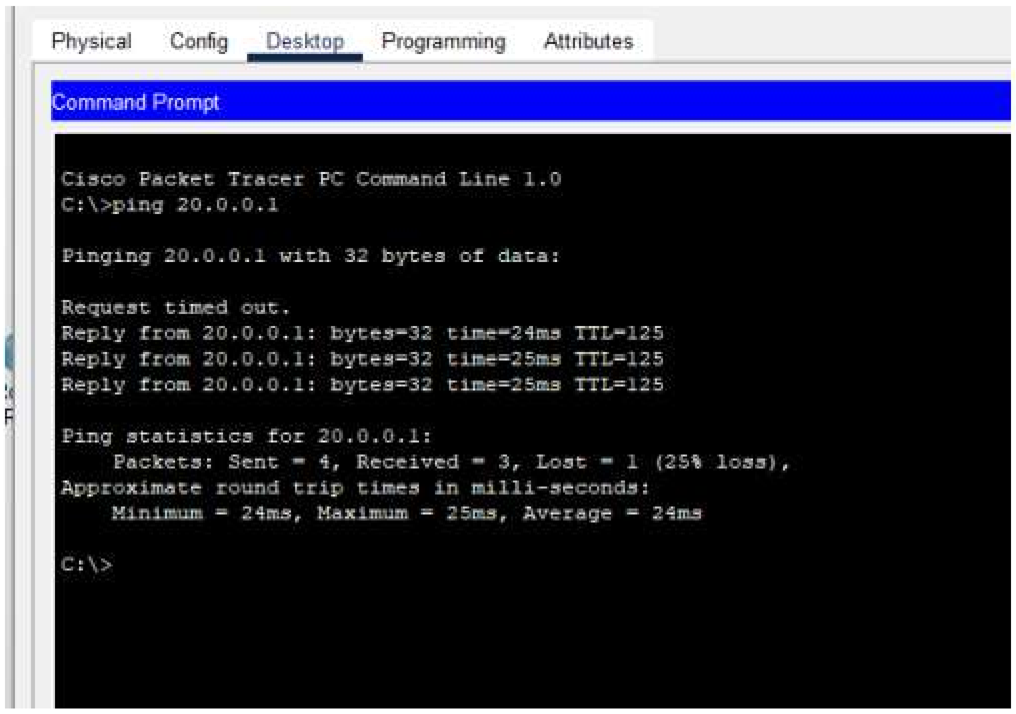


[illegible]

Topology:



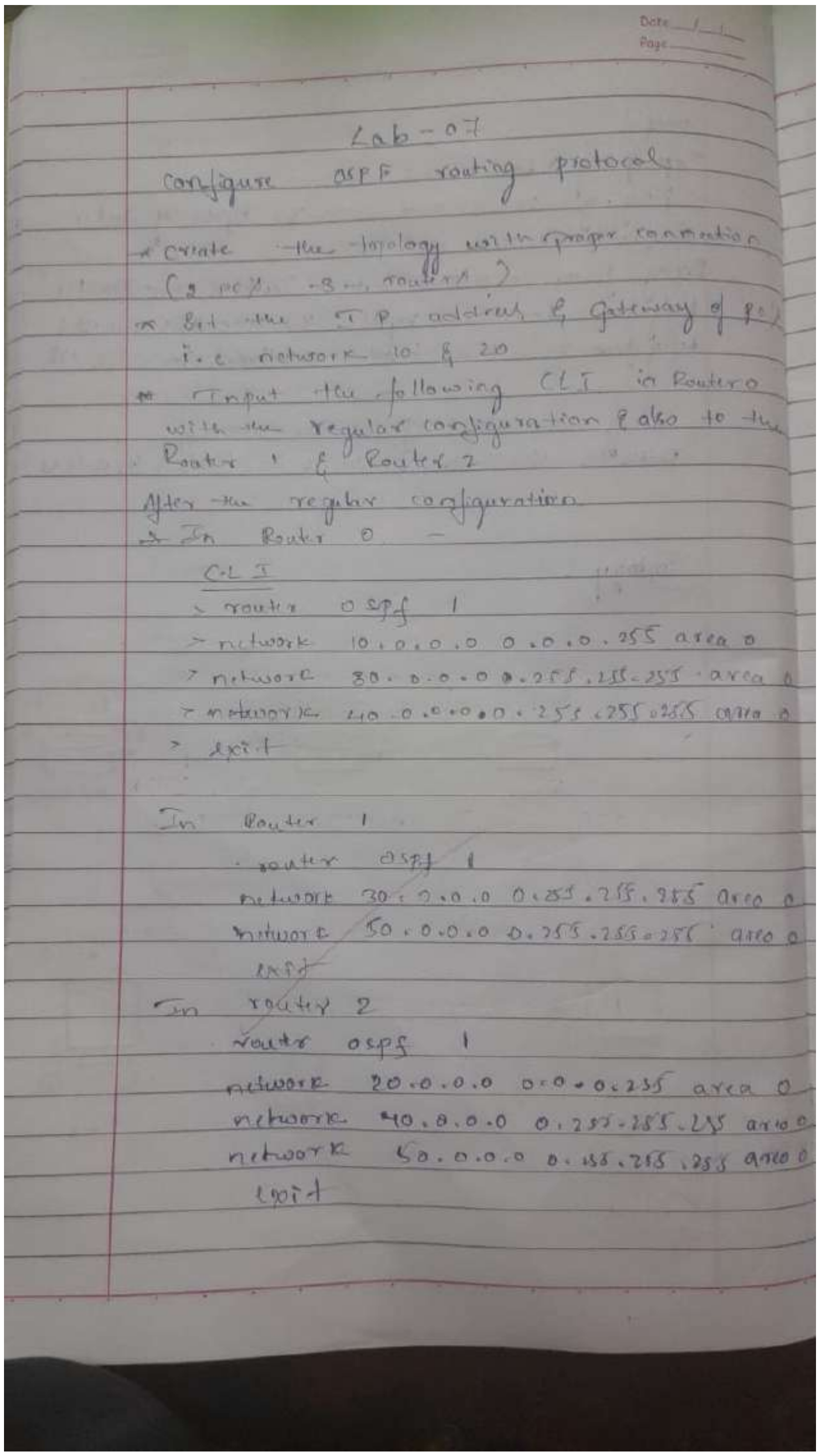
Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.1: bytes=32 time=24ms TTL=125
Reply from 20.0.0.1: bytes=32 time=25ms TTL=125
Reply from 20.0.0.1: bytes=32 time=25ms TTL=125

Ping statistics for 20.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 24ms, Maximum = 25ms, Average = 24ms

C:\>
```

Experiment 6

Configure OSPF routing protocol



Date / /
Page

Lab - 07

Configure OSPF routing protocol.

* Create the topology with proper connection (2 pc's, 3 routers)
* Set the IP address & Gateway of PC i.e network 10 & 20
* Input the following CLI in Router 0 with the regular configuration & also to the Router 1 & Router 2

After the regular configuration
-> In Router 0 -

C.L.I

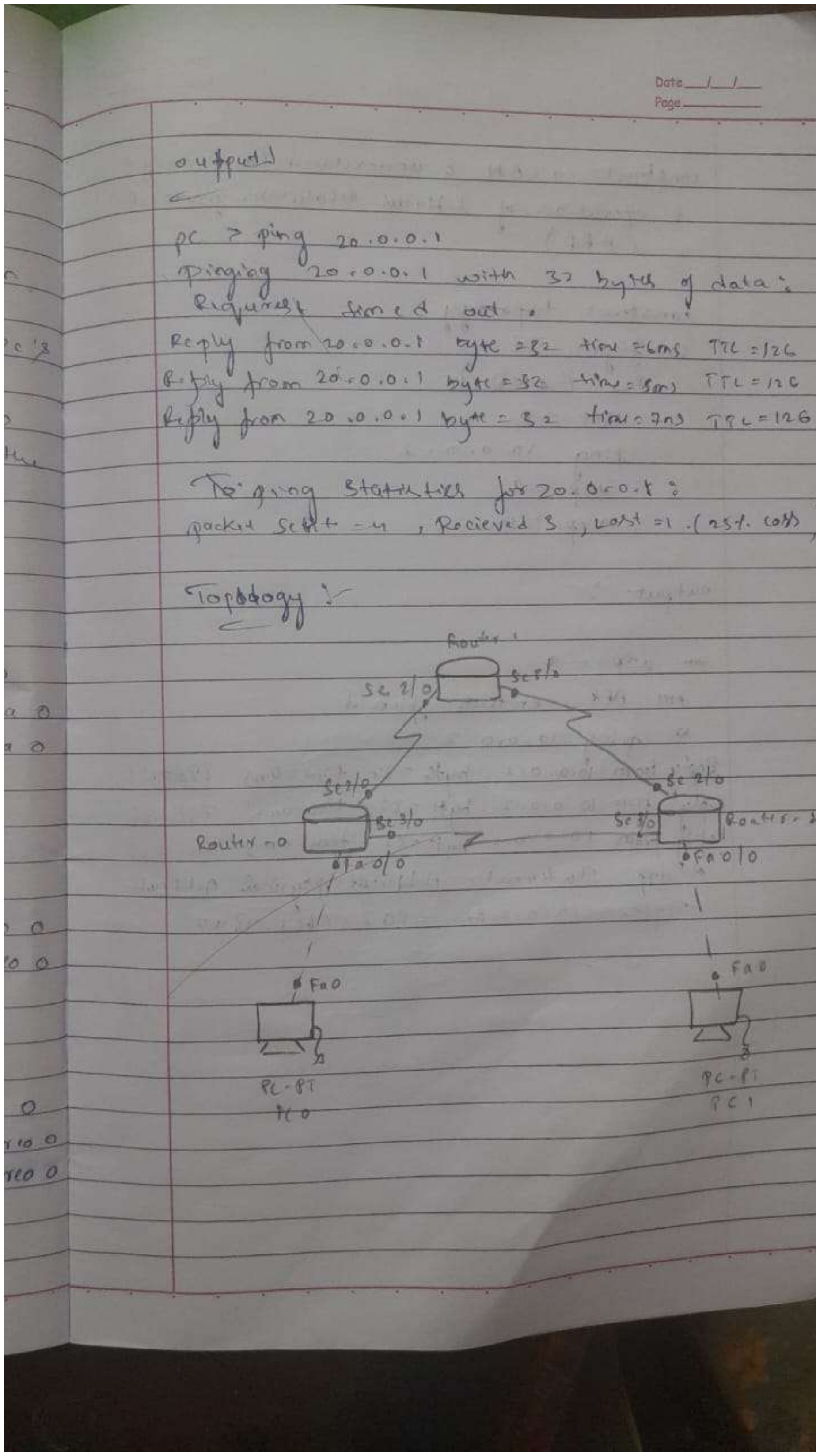
```
> router ospf 1
> network 10.0.0.0 0.0.0.255 area 0
> network 30.0.0.0 0.255.255.255 area 0
> network 40.0.0.0 0.255.255.255 area 0
> exit
```

In Router 1

```
router ospf 1
network 30.0.0.0 0.255.255.255 area 0
network 50.0.0.0 0.255.255.255 area 0
exit
```

In router 2

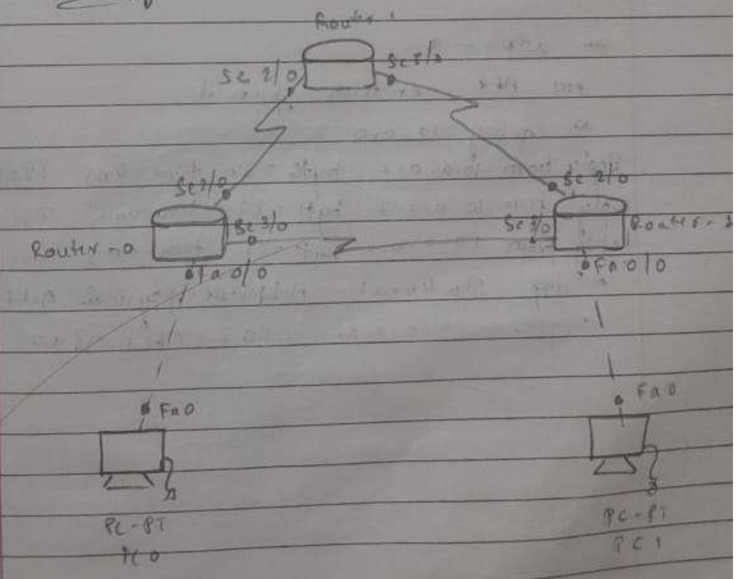
```
router ospf 1
network 20.0.0.0 0.0.0.255 area 0
network 40.0.0.0 0.255.255.255 area 0
network 50.0.0.0 0.255.255.255 area 0
exit
```



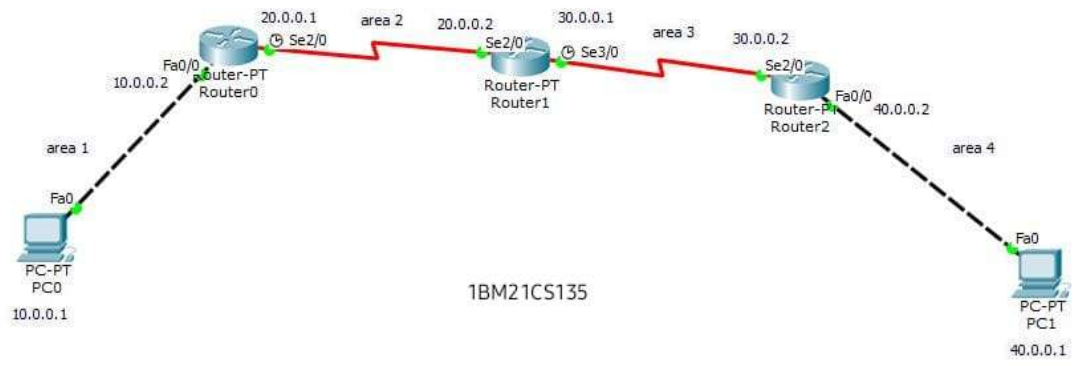
output:-

```
pc > ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:
 Request timed out.
Reply from 20.0.0.1 byte =32 time =6ms TTL =126
Reply from 20.0.0.1 byte = 32 time = 5ms TTL = 126
Reply from 20.0.0.1 byte = 32 time = 7ms TTL = 126

  Ping statistics for 20.0.0.1:
packet Sent = 4 , Recieved 3 , Lost =1 (25% loss)
```

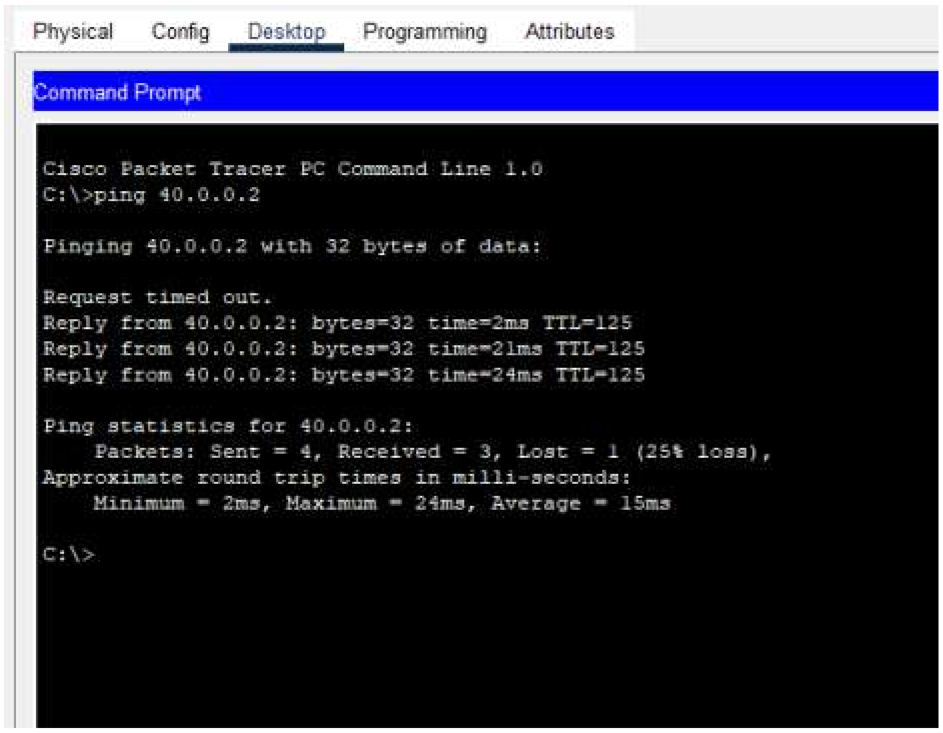
Topology:-



Topology:



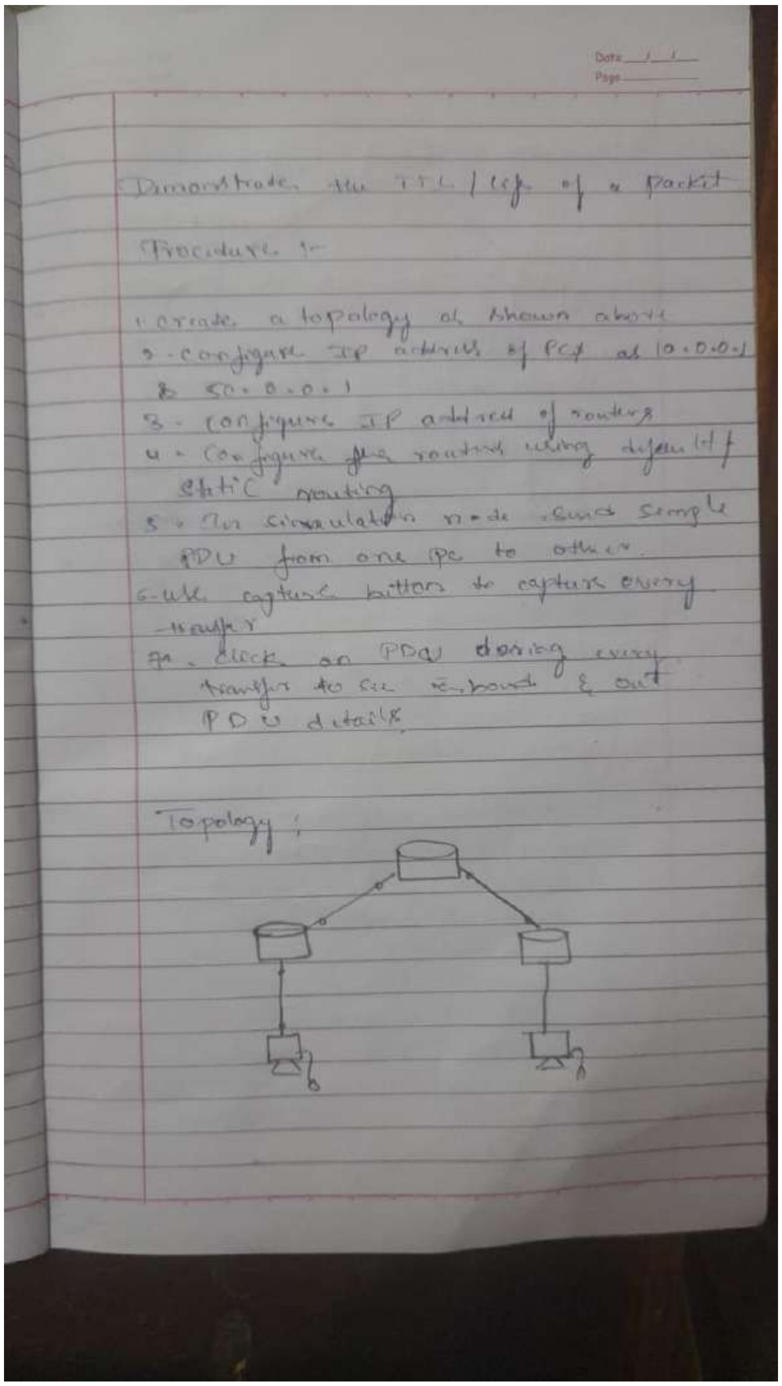
Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 40.0.0.2

Pinging 40.0.0.2 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.2: bytes=32 time=2ms TTL=125
Reply from 40.0.0.2: bytes=32 time=21ms TTL=125
Reply from 40.0.0.2: bytes=32 time=24ms TTL=125

Ping statistics for 40.0.0.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 24ms, Average = 15ms

C:\>
```

Experiment 7

Demonstrate the TTL/ Life of a Packet



Date _/_/_
Page ___

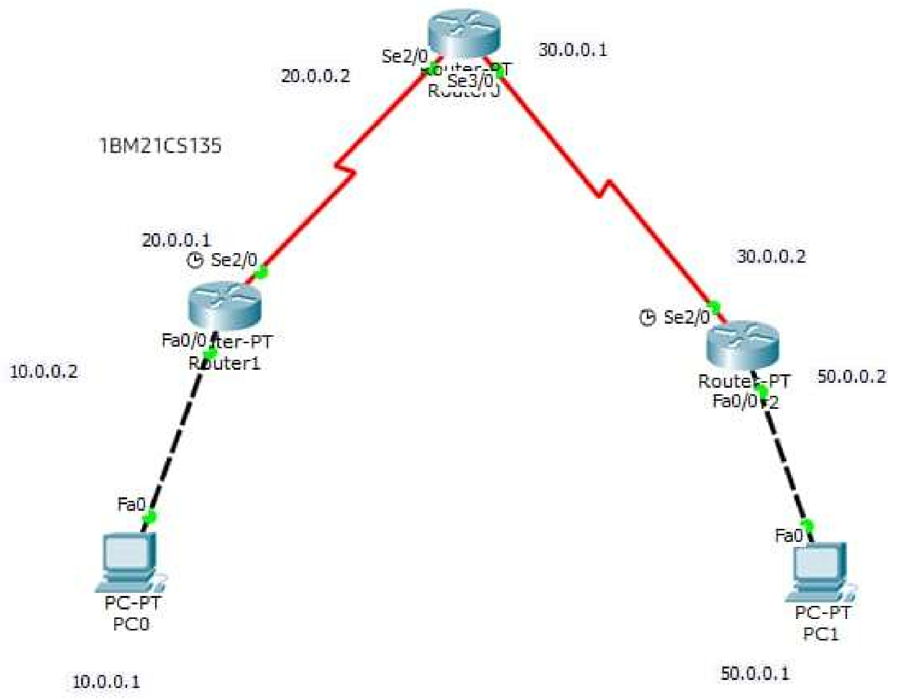
Demonstrate the TTL / life of a packet

Procedure :-

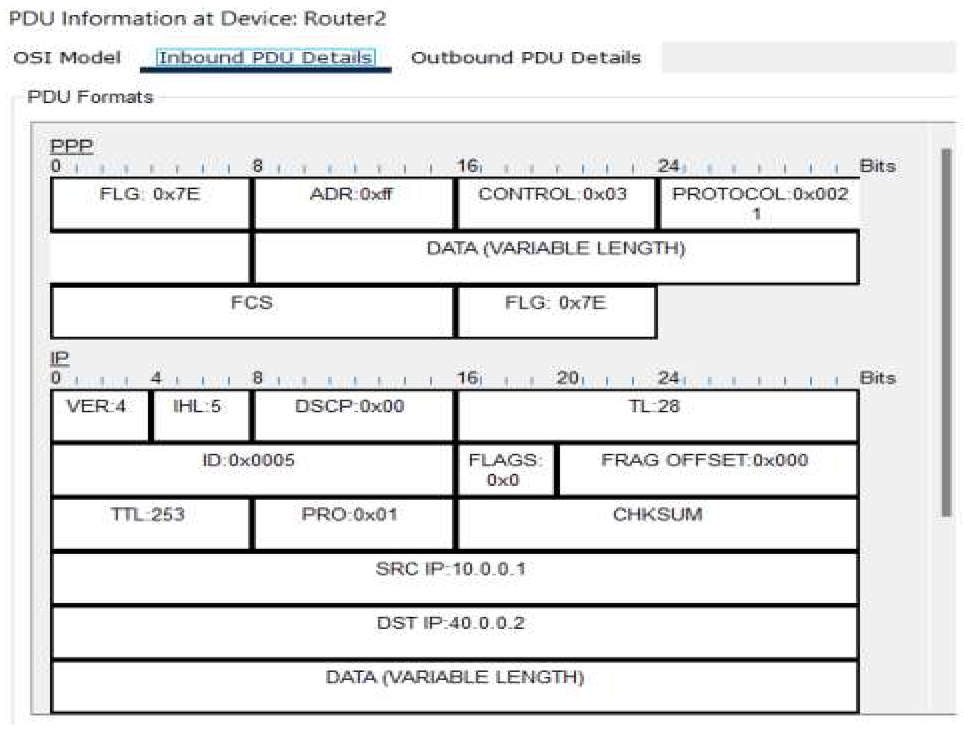
1. create a topology as shown above
2. configure IP address of PCs as 10.0.0.1 & 50.0.0.1
3. configure IP address of routers
4. configure the routers using default / static routing
5. In simulation mode, send simple PDU from one pc to other.
6. use capture button to capture every transfer
7. click on PDU during every transfer to see inbound & out PDU details.

Topology :

Topology:



Output:


PDU Information at Device: Router2

OSI Model | Inbound PDU Details | Outbound PDU Details

PDU Formats

PPP

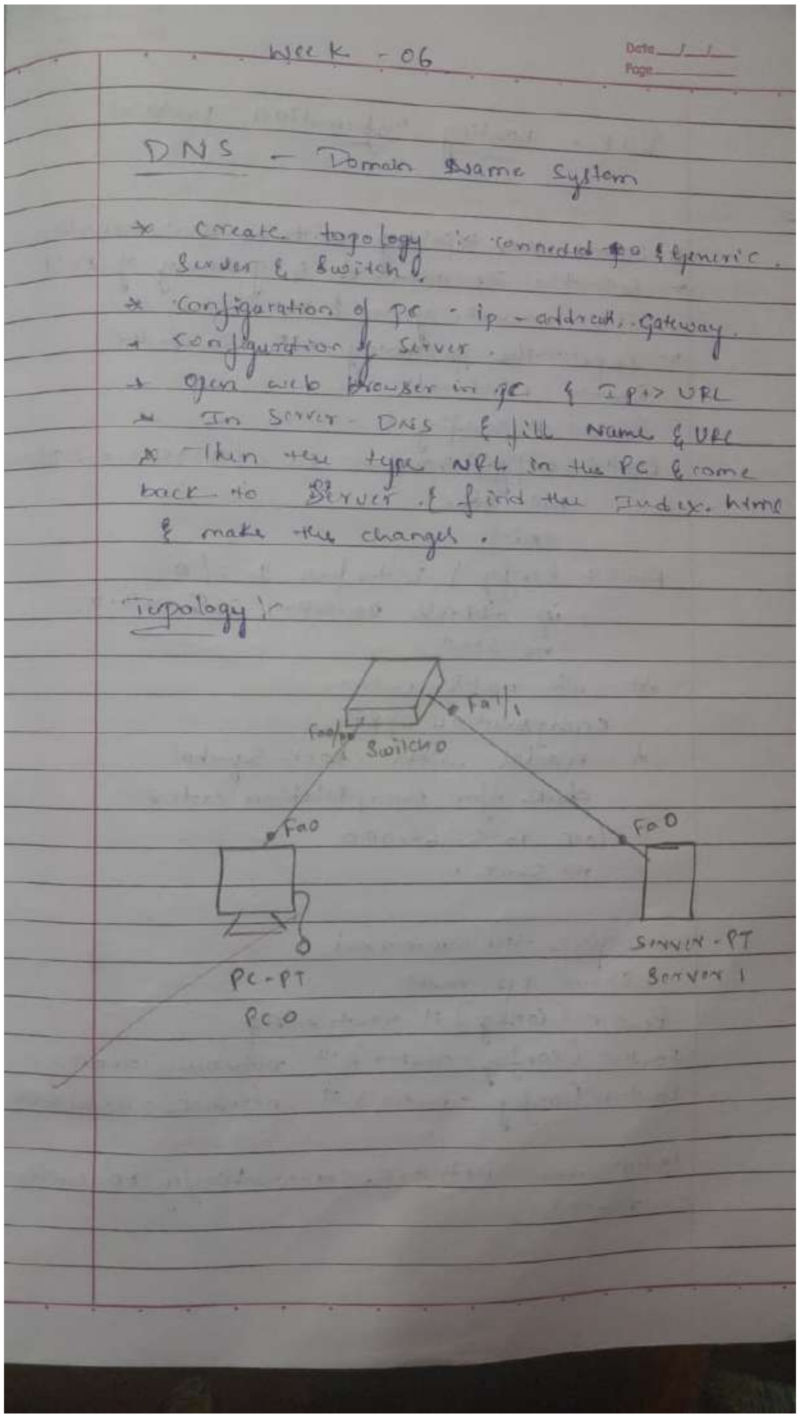
FLG: 0x7E	ADR:0xff	CONTROL:0x03	PROTOCOL:0x0021
	DATA (VARIABLE LENGTH)		
FCS		FLG: 0x7E	

IP

VER:4	IHL:5	DSCP:0x00	TL:28	
ID:0x0005			FLAGS: 0x0	FRAG OFFSET:0x000
TTL:253		PRO:0x01	CHKSUM	
SRC IP:10.0.0.1				
DST IP:40.0.0.2				
DATA (VARIABLE LENGTH)				

Experiment 8

Configure Web Server, DNS within a LAN.



Week - 06

Date __/__/__
Page ____

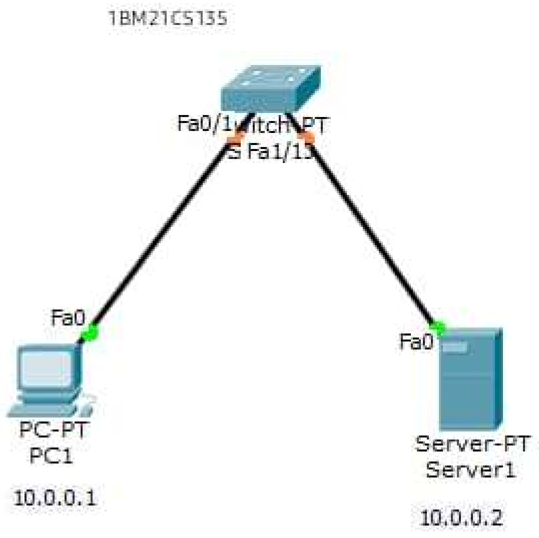
DNS - Domain Name System

* create topology : connected pc0 & generic server & switch 0.
* configuration of pc - ip - address, Gateway.
* configuration of server.
* open web browser in pc & Ip -> URL
* In server - DNS & fill name & URL
* Then the type URL in the PC & come back to server & find the Index. html & make the changes.

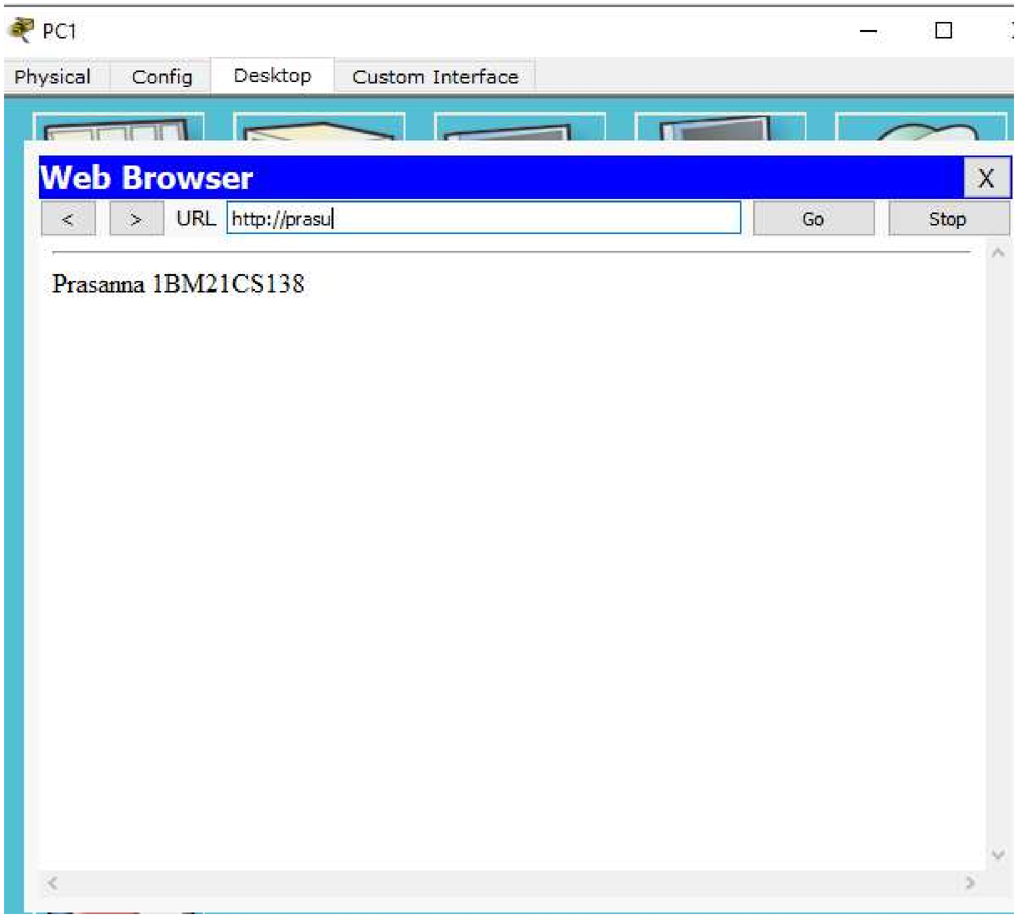
Topology :-

Fa0/1
Fa0/
Switch 0
Fa0
Fa0
PC-PT
PC 0
Server-PT
Server 1

Topology:



Output:

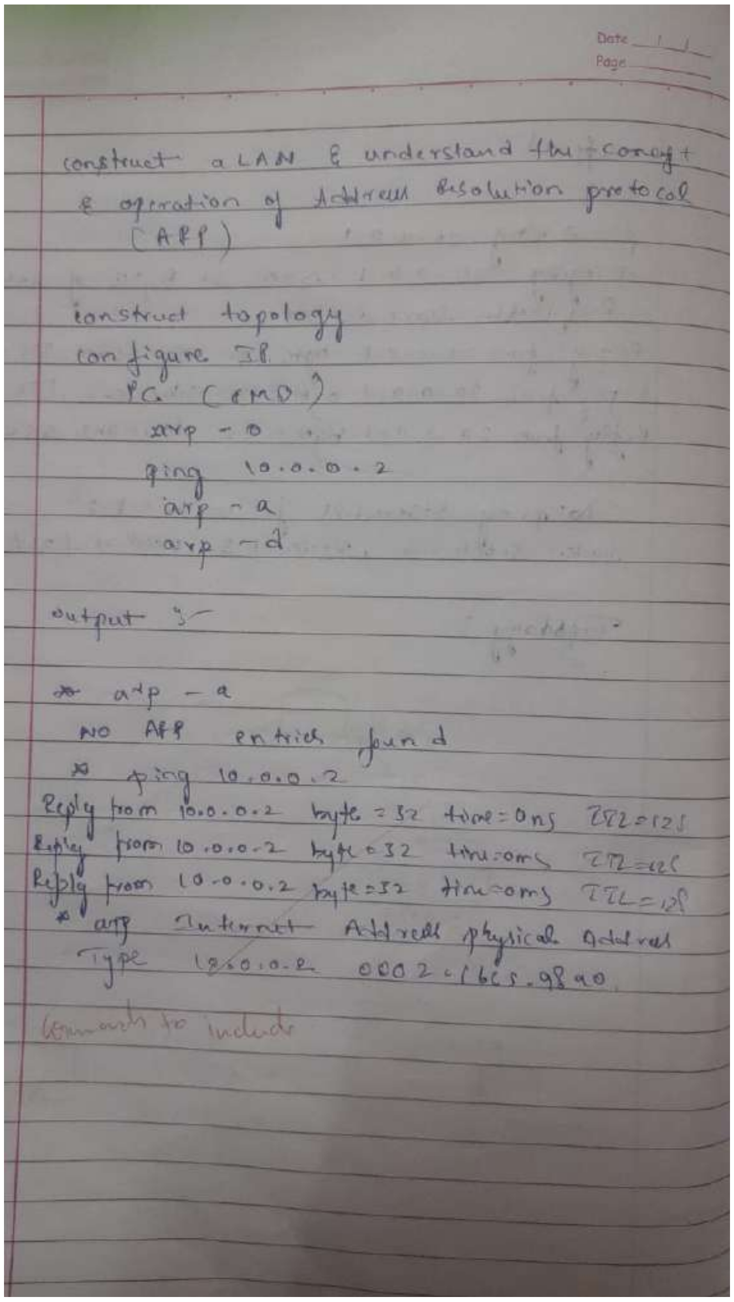


http://prasu

Prasanna 1BM21CS138

Experiment 9

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



Date / /
Page

construct a LAN & understand the concept & operation of Address Resolution protocol (ARP)

construct topology
configure IP
PC (CMD)

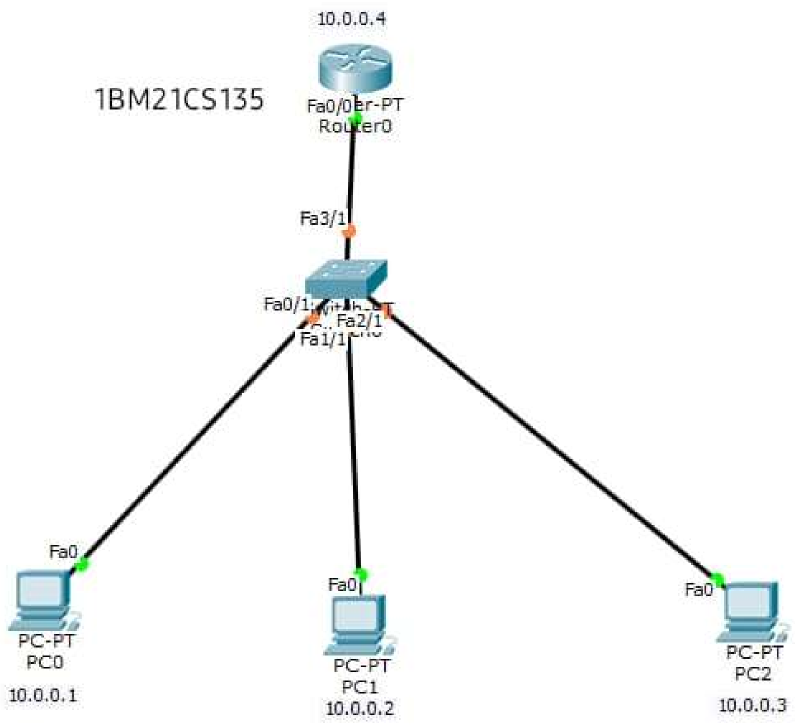
```
arp - a
ping 10.0.0.2
arp - a
arp - d
```

output :-

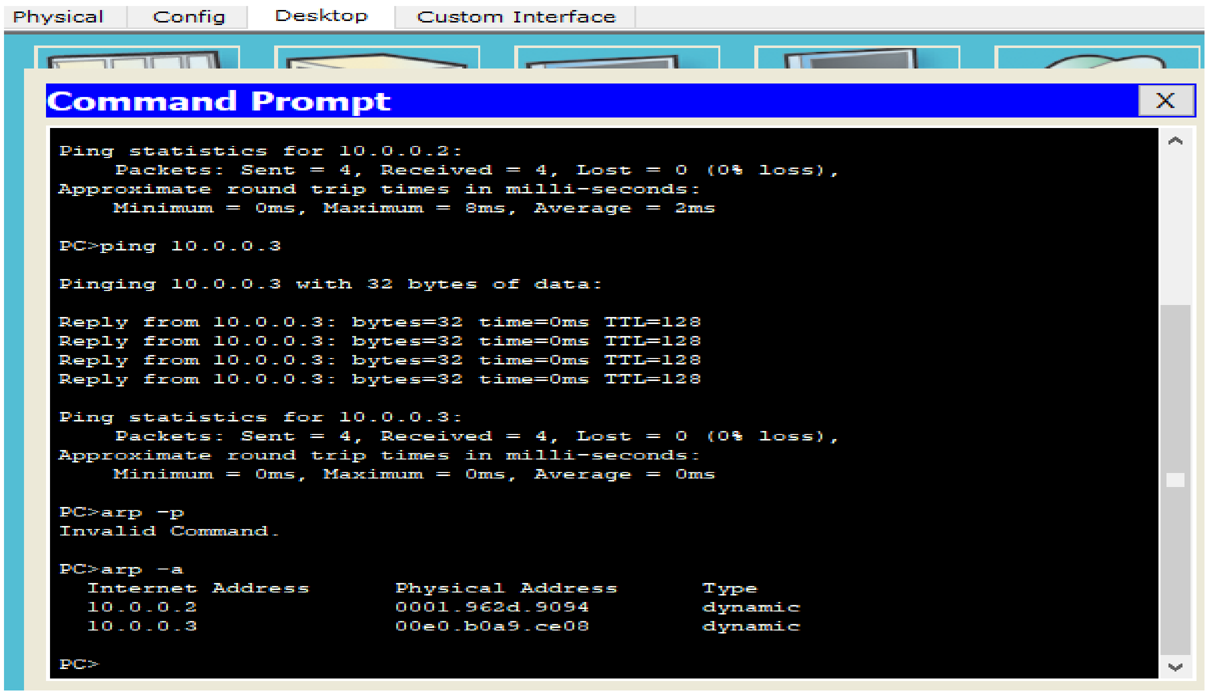
```
* arp - a
  NO ARP entries found
* ping 10.0.0.2
Reply from 10.0.0.2 byte = 32 time = 0ms TTL = 128
Reply from 10.0.0.2 byte = 32 time = 0ms TTL = 128
Reply from 10.0.0.2 byte = 32 time = 0ms TTL = 128
* arp Internet Address physical Address
  Type 10.0.0.2 0002.16c5.9890.
```

Commands to include

Topology:



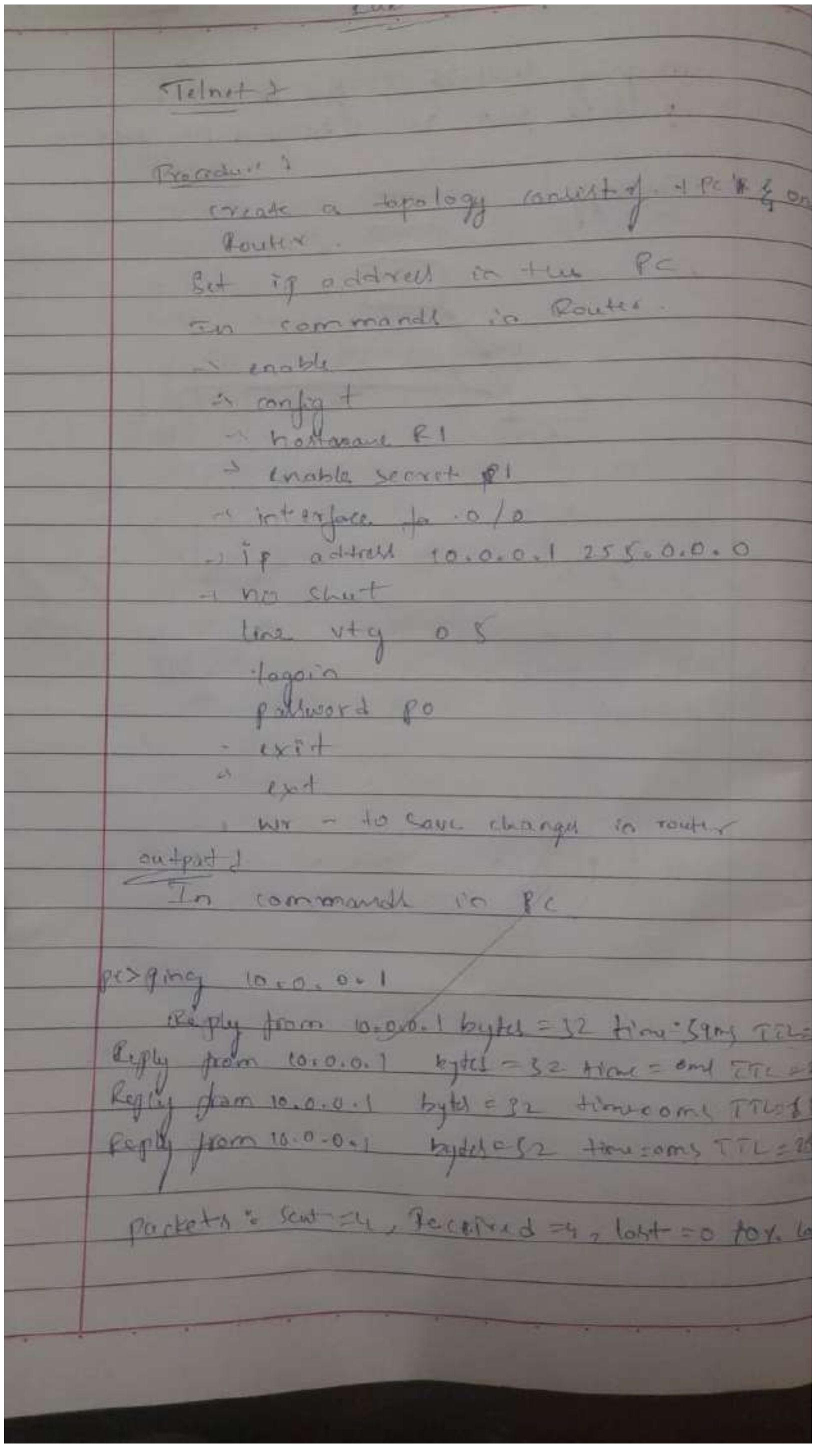
Output:



```
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 = 8ms, Average = 2ms

PC>ping 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data:

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

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

PC>arp -p
Invalid Command.

PC>arp -a
  Internet Address      Physical Address      Type
  10.0.0.2              0001.962d.9094        dynamic
  10.0.0.3              00e0.b0a9.ce08        dynamic

PC>
```

Experiment 10

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



Telnet :

Procedure :

create a topology consisting 1 PC & one Router.

Set ip address in the PC

In commands in Router.

```
-> enable
-> config t
-> hostname R1
-> enable secret p1
-> interface fa 0/0
-> ip address 10.0.0.1 255.0.0.0
-> no shut
   line vty 0 5
   login
   password p0
-  exit
-> exit
.  wr  - to save changes in router
```

output :

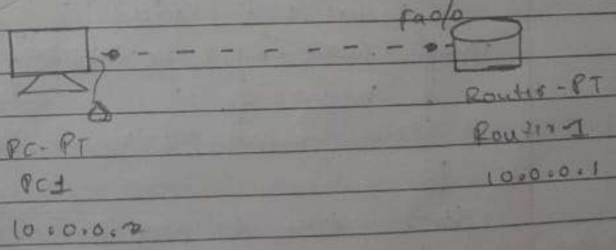
In commands in PC

```
pc>ping 10.0.0.1
    Reply from 10.0.0.1 bytes = 32 time=39ms TTL=
Reply from 10.0.0.1 bytes = 32 time = 0ms TTL =
Reply from 10.0.0.1 bytes = 32 time=0ms TTL=
Reply from 10.0.0.1 bytes=32 time=0ms TTL=

packets : Sent = 4, Received = 4, lost = 0 (0% lo
```

Date __/__/__
Page ______

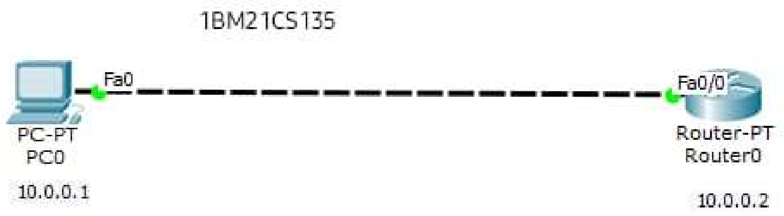
```
PC > telnet 10.0.0.1
: Trying 10.0.0.1 ... open
User Access verification
password :
  R1 > enable
 password :
        r1# : show ip route
codes : C - connected , S - static , I - IGRP , R - RIP ,
 M - Mobile , B BGP , D - EIGRP , EX - EIGRP ,
  O - OSPF , IA - OSPF intern area
 N1 - OSPF NSSFA external type
 area
     * - candidate default
     P - periodic download static rout
 Gateway of last report is not set
 C  10.0.0.0/0 is directly connected Fa 0/0
```

Topology :

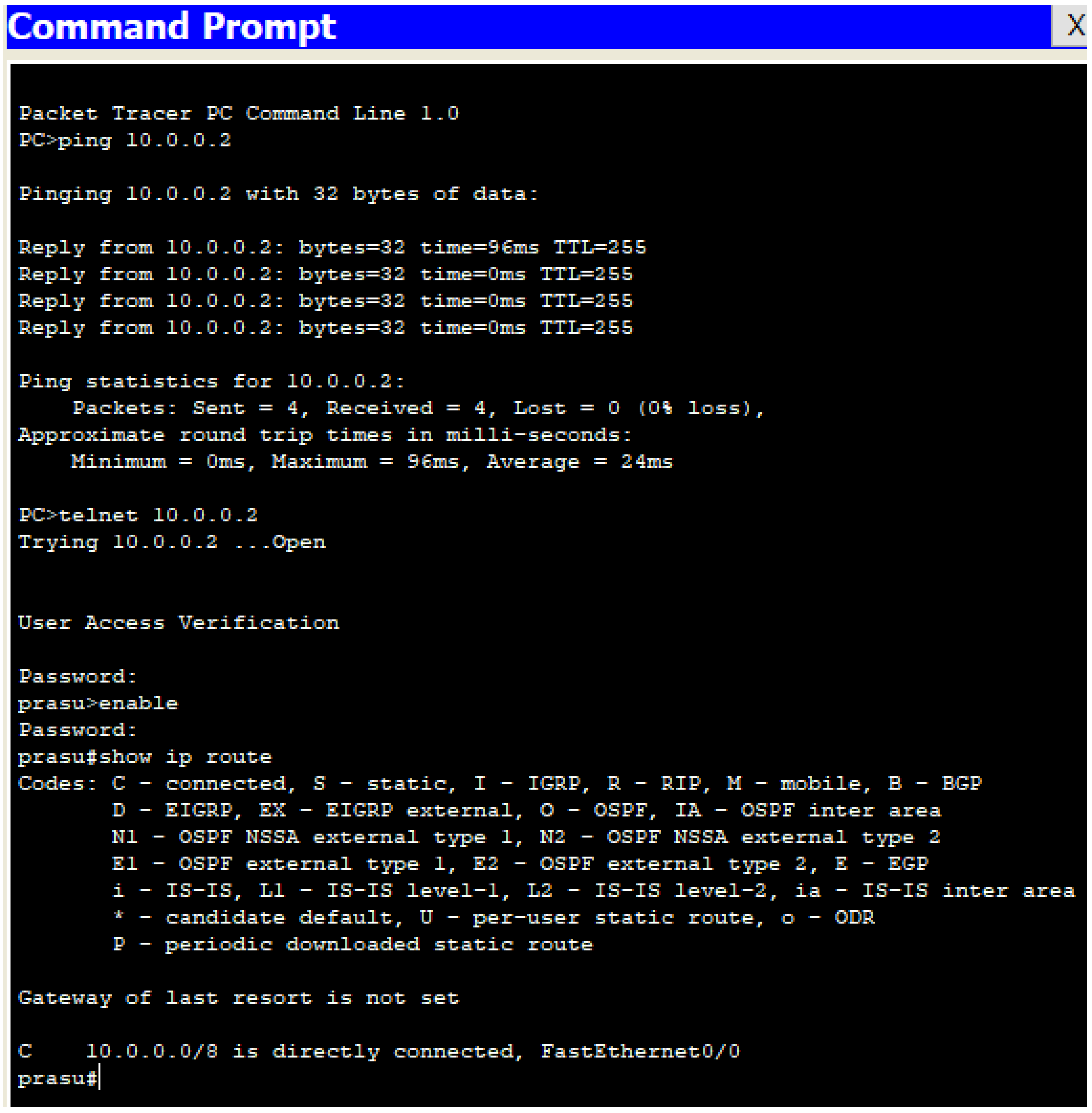


[illegible]

Topology:



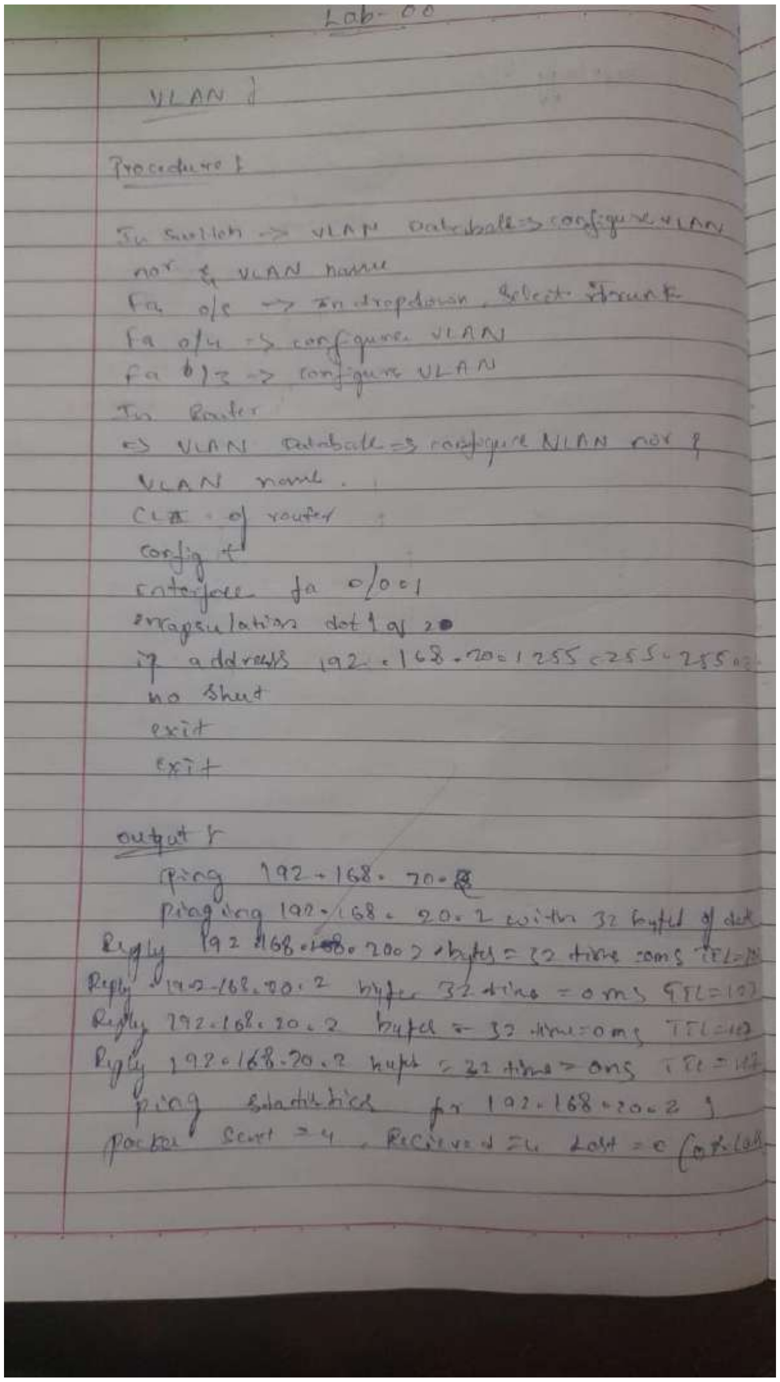
Output:



```
Command Prompt

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=96ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255
Reply from 10.0.0.2: bytes=32 time=0ms TTL=255

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 96ms, Average = 24ms

PC>telnet 10.0.0.2
Trying 10.0.0.2 ...Open


User Access Verification

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

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
prasu#
```

Experiment 11

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



Lab-00

VLAN :

Procedure :

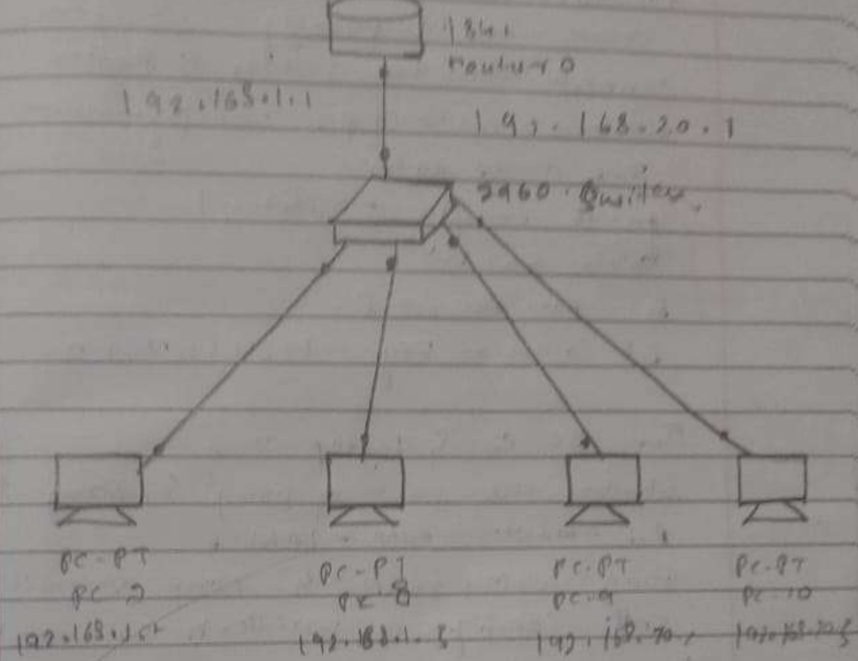
In Switch -> VLAN Database -> configure VLAN
no & VLAN name
Fa 0/5 -> In dropdown, Select Trunk
Fa 0/4 -> configure VLAN
Fa 0/3 -> configure VLAN
In Router
=> VLAN Database -> configure VLAN no &
VLAN name.
CLI of router

```
config t
interface fa 0/0.1
encapsulation dot1q 20
ip address 192.168.20.1 255.255.255.0
no shut
exit
exit
```

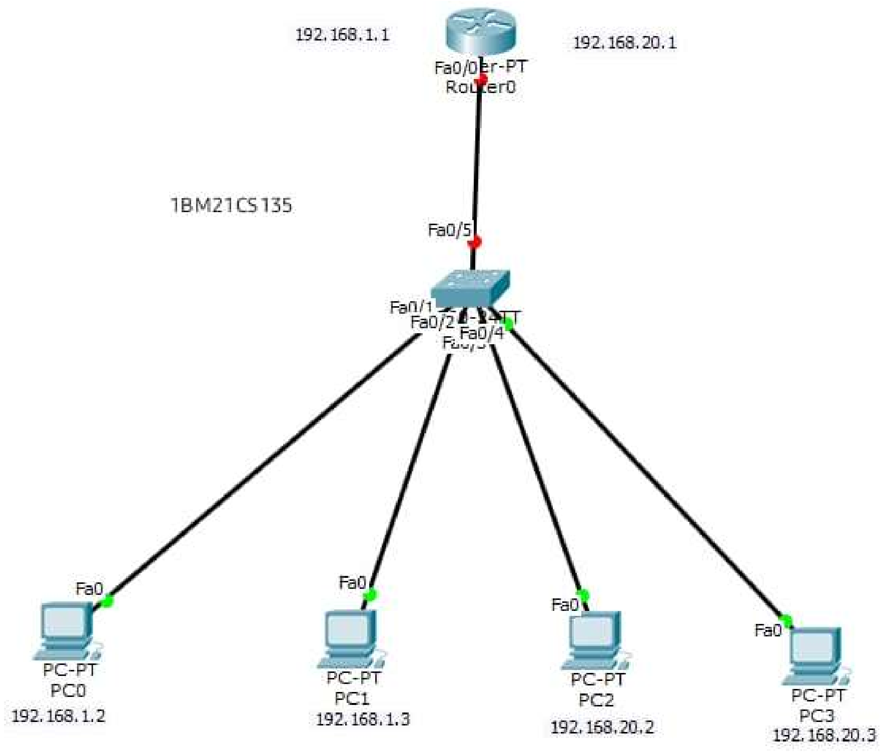
Output :

```
ping 192.168.20.2
pinging 192.168.20.2 with 32 bytes of data
Reply 192.168.20.2 bytes=32 time=0ms TTL=128
Reply 192.168.20.2 bytes 32 time = 0ms TTL=127
Reply 192.168.20.2 bytes = 32 time=0ms TTL=127
Reply 192.168.20.2 bytes = 32 time = 0ms TTL=127
ping statistics for 192.168.20.2 :
Packet Sent = 4, Received = 4, Lost = 0 (0% loss
```

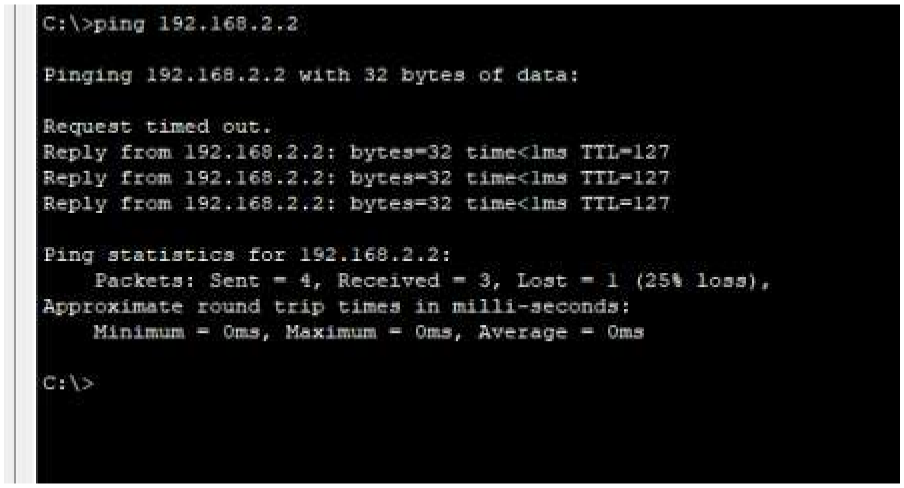
Topology:


1841
Router0
192.168.1.1
192.168.20.1
2960 Switch
PC-PT
PC 7
192.168.1.5
PC-PT
PC 8
192.168.1.5
PC-PT
PC 9
192.168.20.
PC-PT
PC 10
192.168.20.5

Topology:



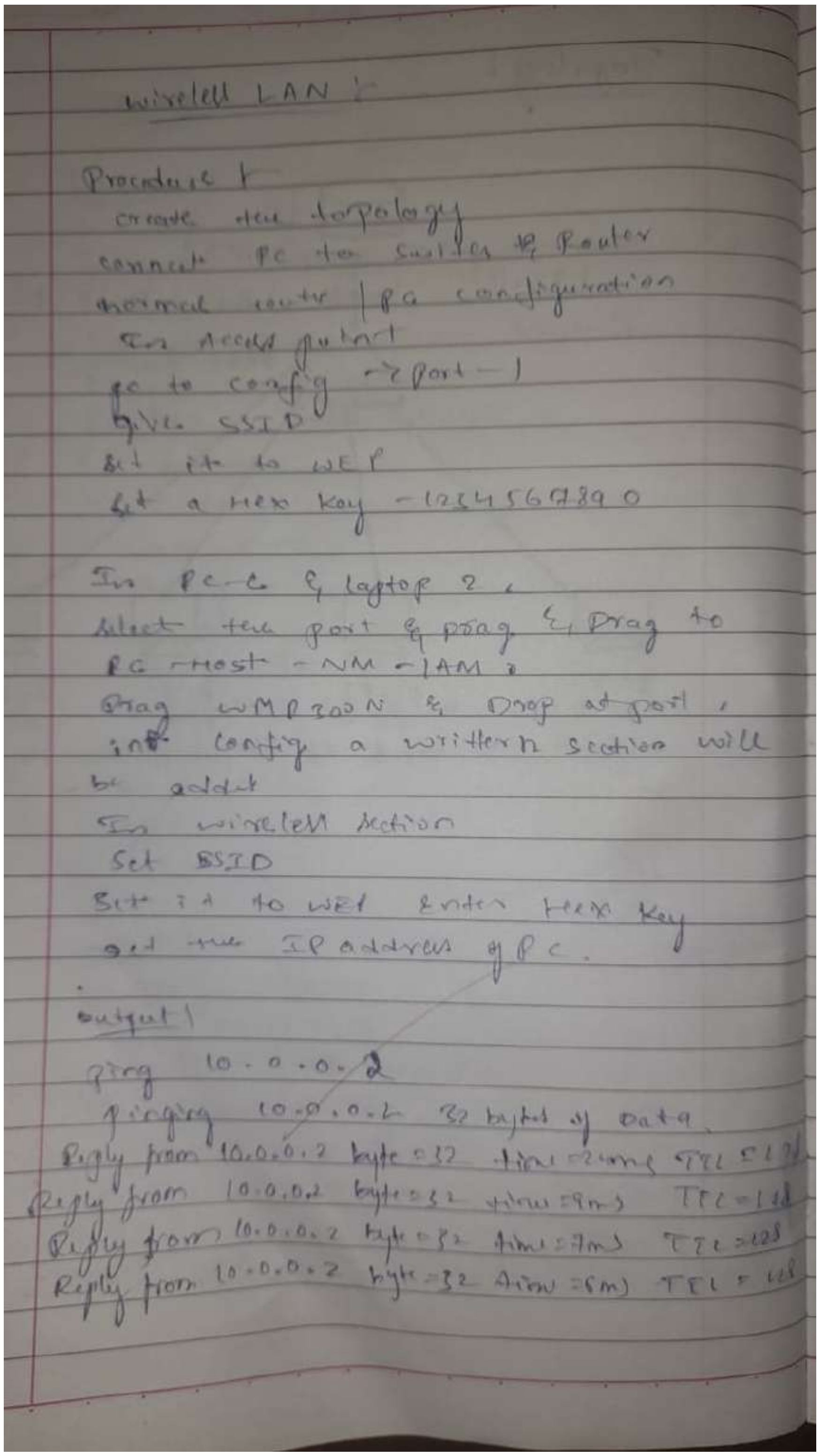
Output:



```
C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Request timed out.
Reply from 192.168.2.2: bytes=32 time<1ms TTL=127
Reply from 192.168.2.2: bytes=32 time<1ms TTL=127
Reply from 192.168.2.2: bytes=32 time<1ms TTL=127

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

Experiment 12

To construct a WLAN and make the nodes communicate wirelessly



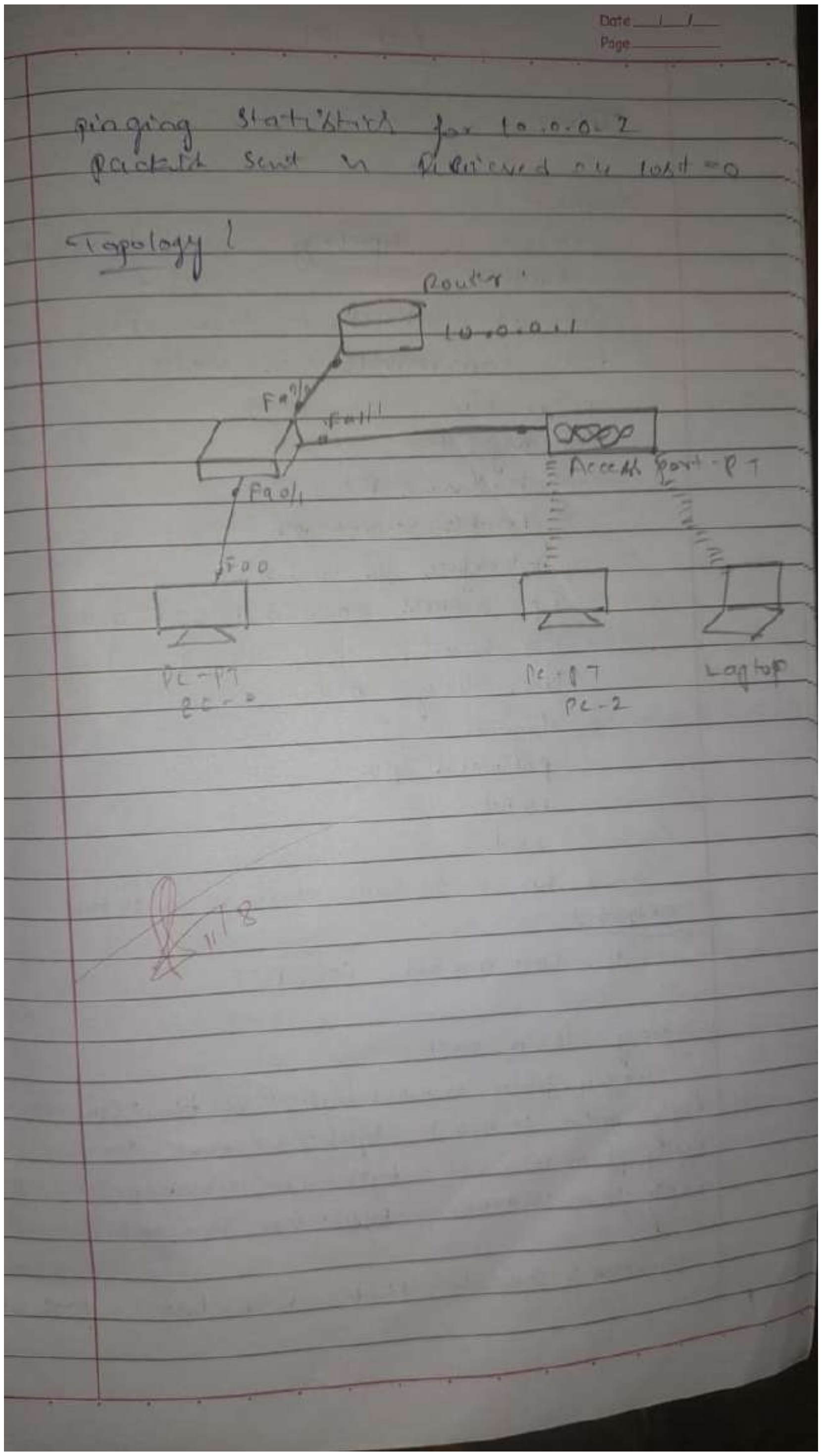
wireless LAN :

Procedure :
create the topology
connect PC to switches & Router
normal router | PC configuration
In Access point
go to config -> port - 1
give SSID
set it to WEP
set a Hex key - 1234567890

In PC-0 & laptop 2,
select the port & drag & drag to
PC - Host - NM - 1AM :
Drag WMP300N & Drop at port,
in config a wireless section will
be added
In wireless section
Set SSID
Set it to WEP enter Hex Key
set the IP address of PC.

output :

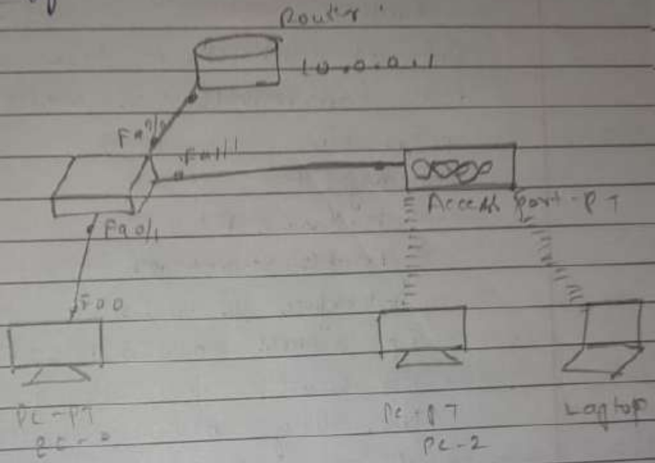
```
ping 10.0.0.2
  pinging 10.0.0.2 32 bytes of Data.
Reply from 10.0.0.2 byte=32 time=2ms TTL=128
Reply from 10.0.0.2 byte=32 time=9ms TTL=128
Reply from 10.0.0.2 byte=32 time=7ms TTL=128
Reply from 10.0.0.2 byte=32 time=5ms TTL=128
```



Date__/__/__
Page____

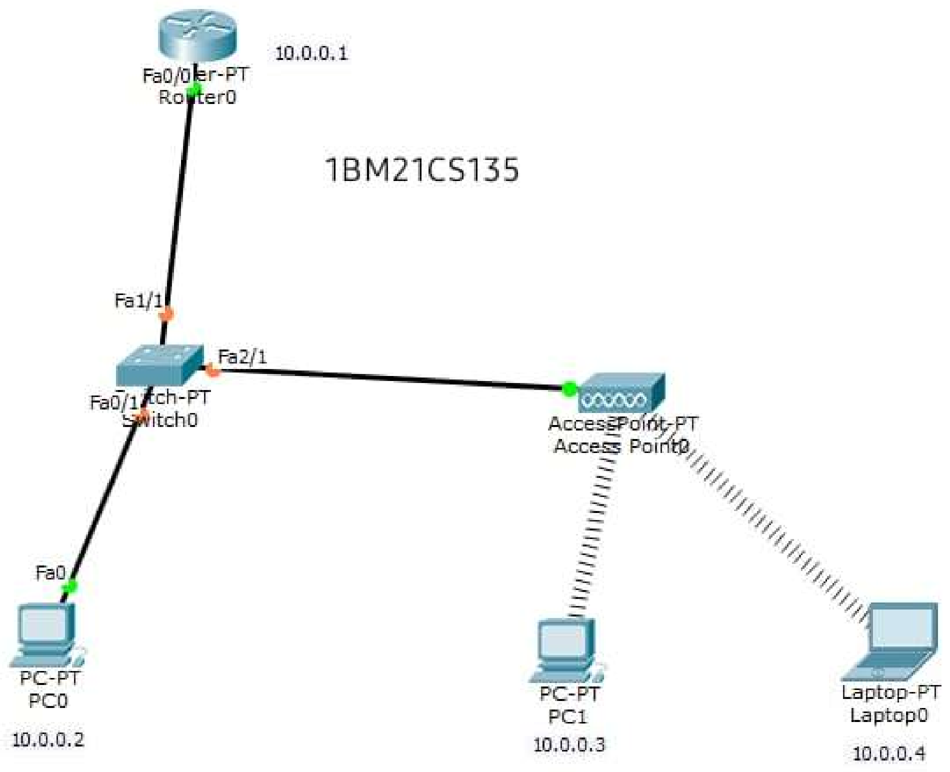
pinging statistics for 10.0.0.2
packets sent n Recieved n lost = 0

Topology :

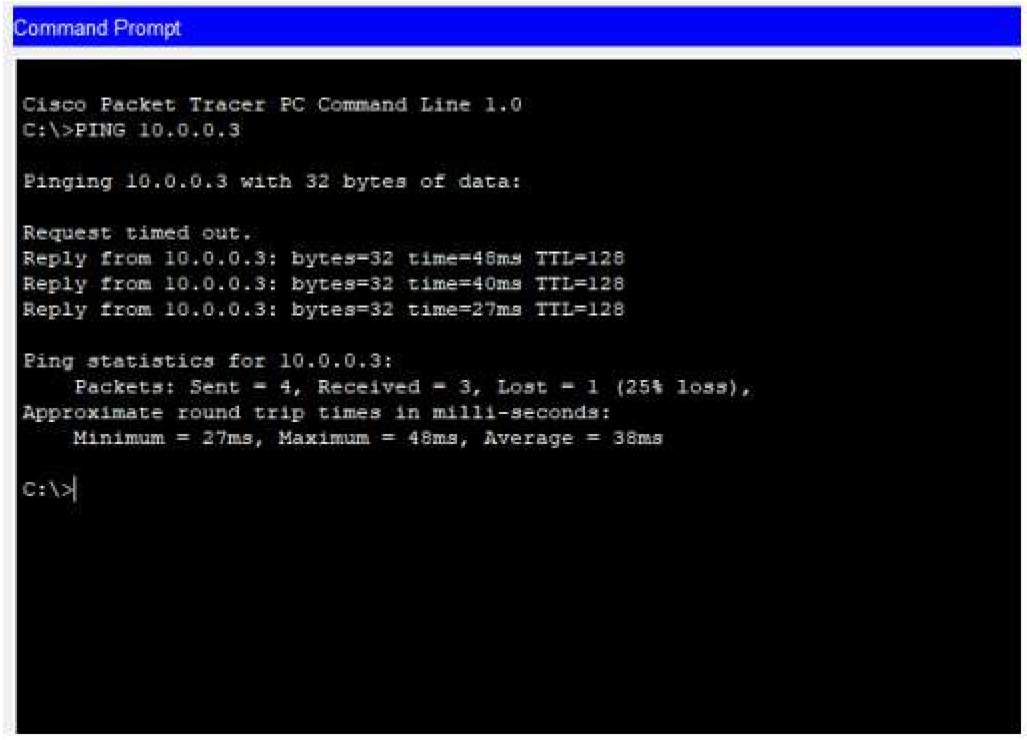


11/8

Topology:



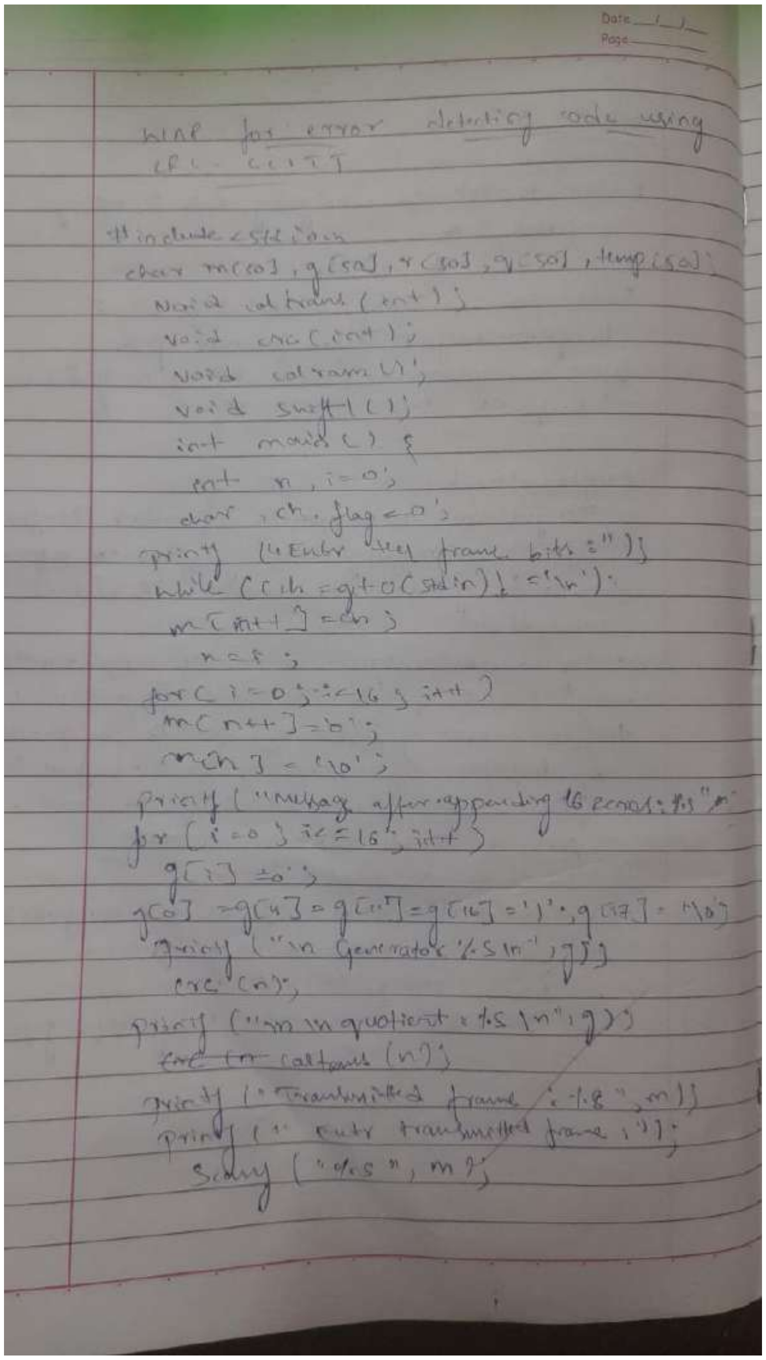
Output:



```
Cisco Packet Tracer PC Command Line 1.0
C:\>PING 10.0.0.3

Pinging 10.0.0.3 with 32 bytes of data:

Request timed out.
Reply from 10.0.0.3: bytes=32 time=48ms TTL=128
Reply from 10.0.0.3: bytes=32 time=40ms TTL=128
Reply from 10.0.0.3: bytes=32 time=27ms TTL=128

Ping statistics for 10.0.0.3:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 27ms, Maximum = 48ms, Average = 38ms

C:\>
```

Experiment 13

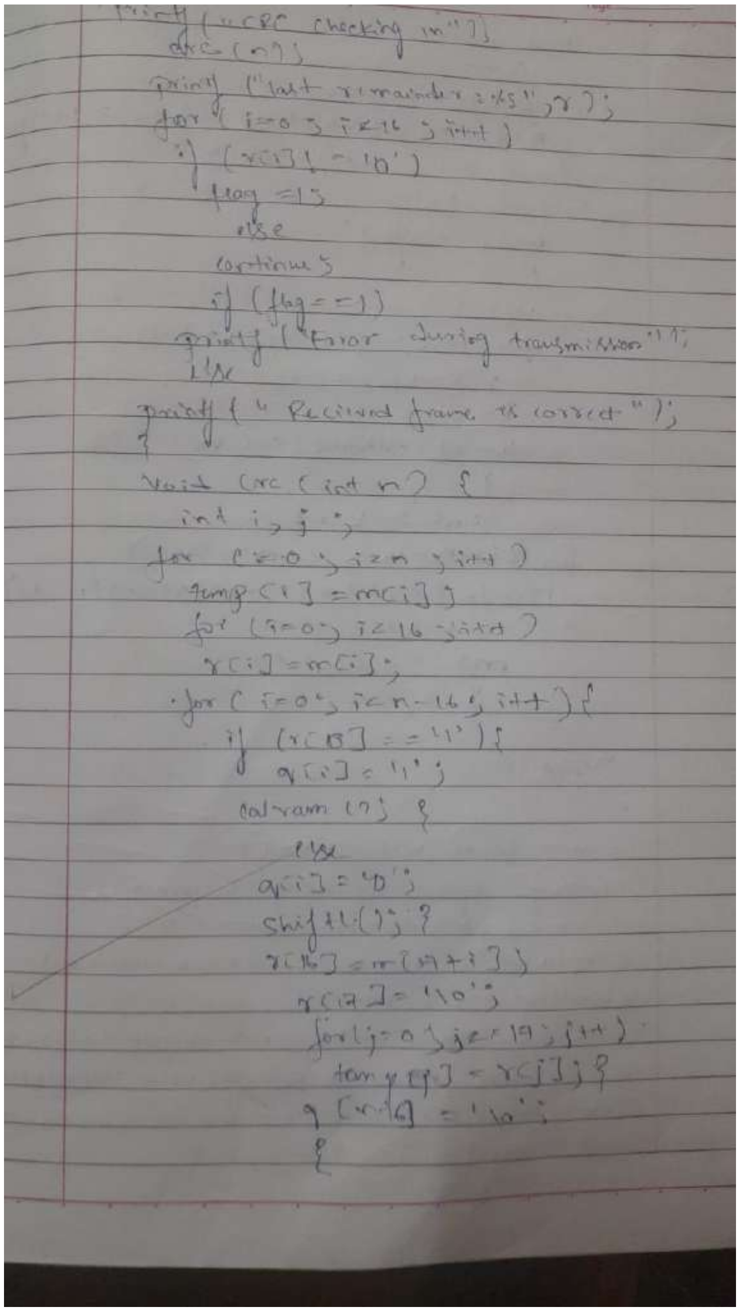
Write a program for error detecting code using CRC-CCITT (16-bits).



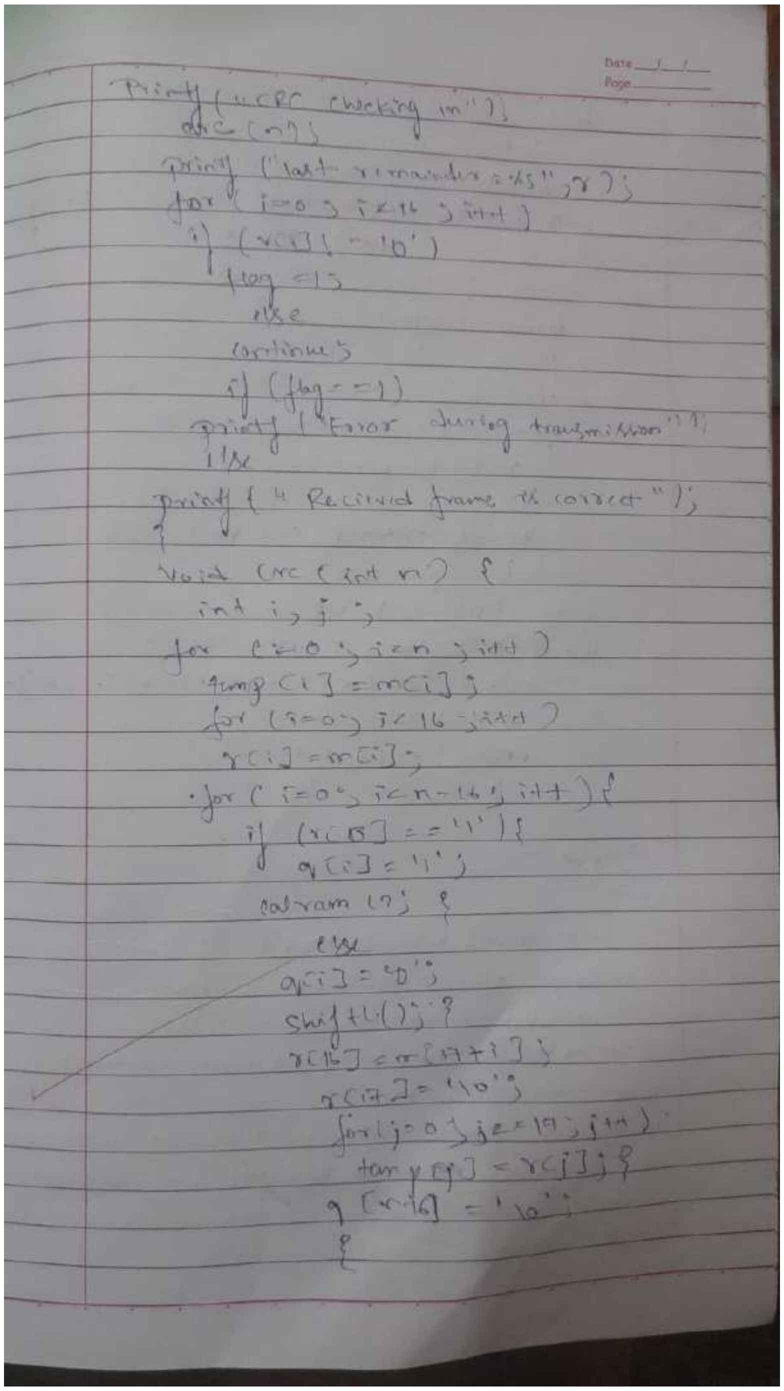
Date __/__/__
Page ____

WAP for error detecting code using CRC - CCITT

```
#include <stdio.h>
char m[50], g[50], r[50], q[50], temp[50];
  void caltrans (int);
   void crc (int);
    void calram ();
    void shift1 ();
    int main () {
     int n, i=0;
     char ch, flag=0;
  printf ("Enter the frame bits :");
  while ((ch=getc(stdin))!='\n')
   m[n++]=ch;
     n=i;
 for ( i=0; i<16; i++)
  m[n++]='0';
  m[n] = '\0';
 printf ("message after appending 16 zeros: %s", m
for ( i=0 ; i<=16; i++)
   g[i] ='0';
g[0] =g[4]=g[11]=g[16] ='1'; g[17] ='\0';
 printf ("\n Generator %s\n", g);
   crc (n);
printf ("\n\n quotient : %s \n", q);
 ~~crc (n~~ caltrans (n);
 printf ("Transmitted frame : %s", m);
  printf (" enter transmitted frame :");
    scanf ("%s", m);
```

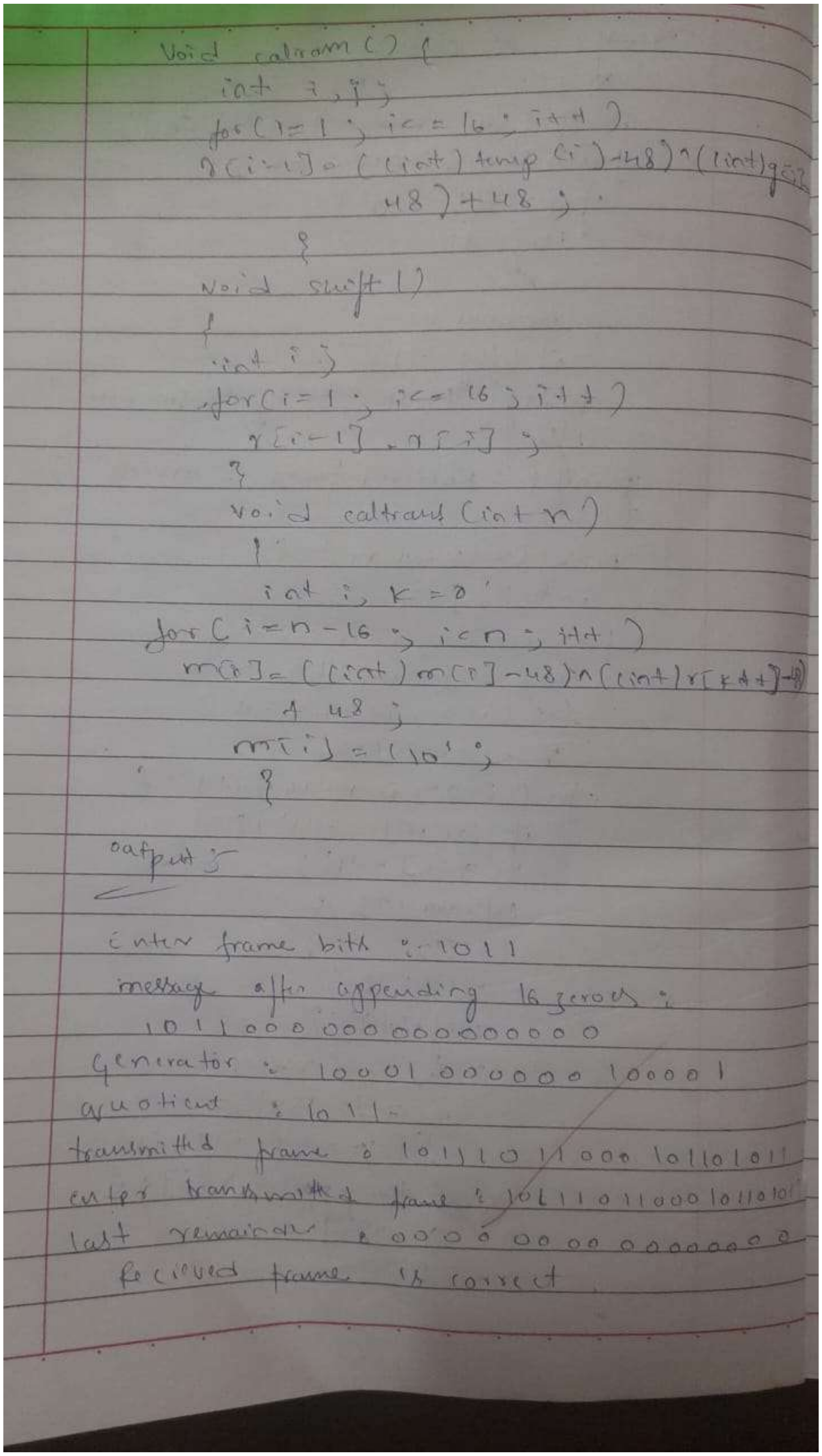


```
printf ("\nCRC checking \n");
  crc (n);
 printf ("last remainder : %s", r);
 for ( i=0 ; i<16 ; i++ )
   if (r[i] != '0')
    flag =1;
      else
    continue;
    if (flag==1)
  printf ("Error during transmission");
  else
printf (" Recieved frame is correct ");
}

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



Date __/__/__
Page ______

```
Printf ("CRC checking \n");
  crc (n);
 printf ("last remainder = %s", r);
 for ( i=0 ; i<16 ; i++ )
  if (r[i] != '0')
   flag =1;
    else
   continue;
   if (flag = =1)
  printf ("Error during transmission");
  else
 printf ( " Recived frame is correct ");
}

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

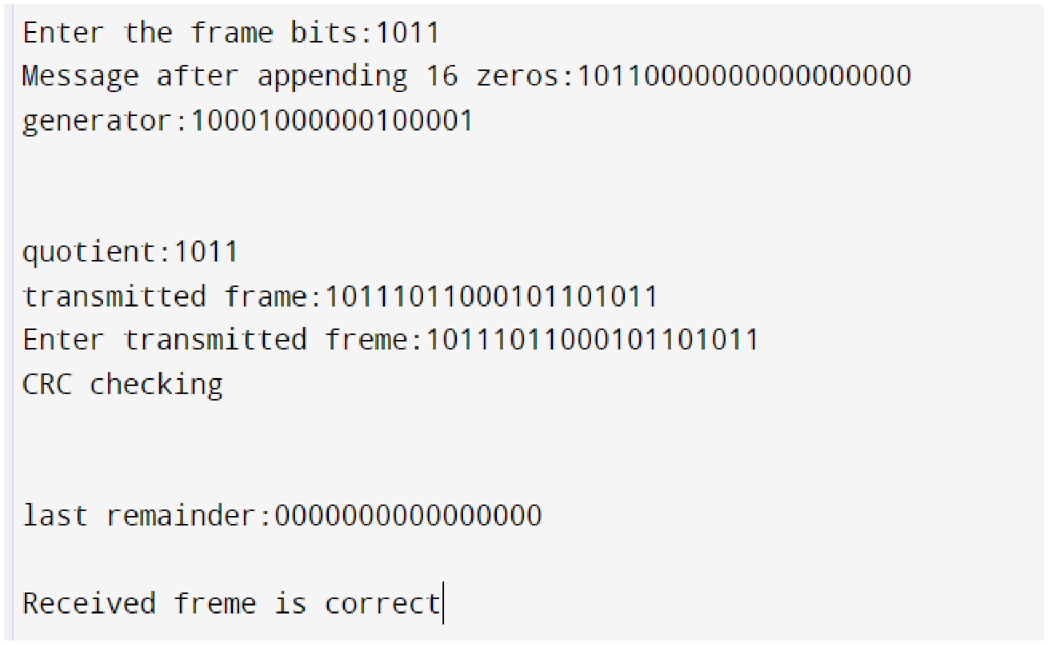


```
Void calram() {
    int i, j;
    for(i=1; i<=16; i++)
    r[i-1] = ((int)temp[i]-48)^((int)g[i]-
                48)+48;
        }
    Void shift()
    {
    int i;
    for(i=1; i<=16; i++)
      r[i-1] = r[i];
    }
    void caltrans(int n)
     {
      int i, k=0;
  for(i=n-16; i<n; i++)
    m[i] = ((int)m[i]-48)^((int)r[k++]-48)
        + 48;
    m[i] = '\0';
     }
```

output :

```
enter frame bits : 1011
message after appending 16 zeros :
  10110000000000000000
Generator : 10001000000100001
quotient : 1011
transmitted frame : 10111011000101101011
enter transmitted frame : 10111011000101101011
last remainder : 0000000000000000
  Received frame is correct
```

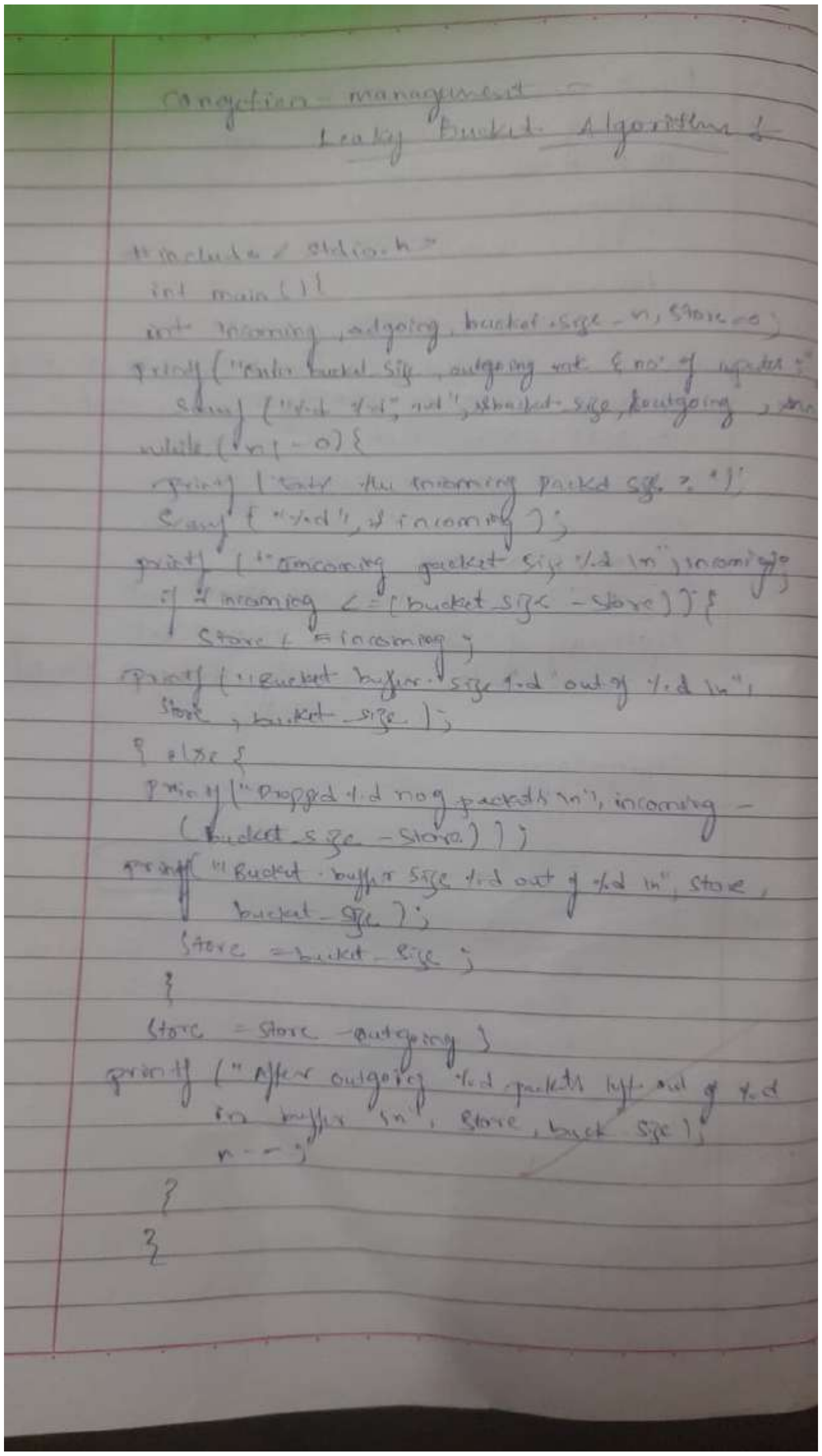
Output:



```
Enter the frame bits:1011
Message after appending 16 zeros:10110000000000000000
generator:10001000000100001


quotient:1011
transmitted frame:10111011000101101011
Enter transmitted freme:10111011000101101011
CRC checking


last remainder:0000000000000000

Received freme is correct
```

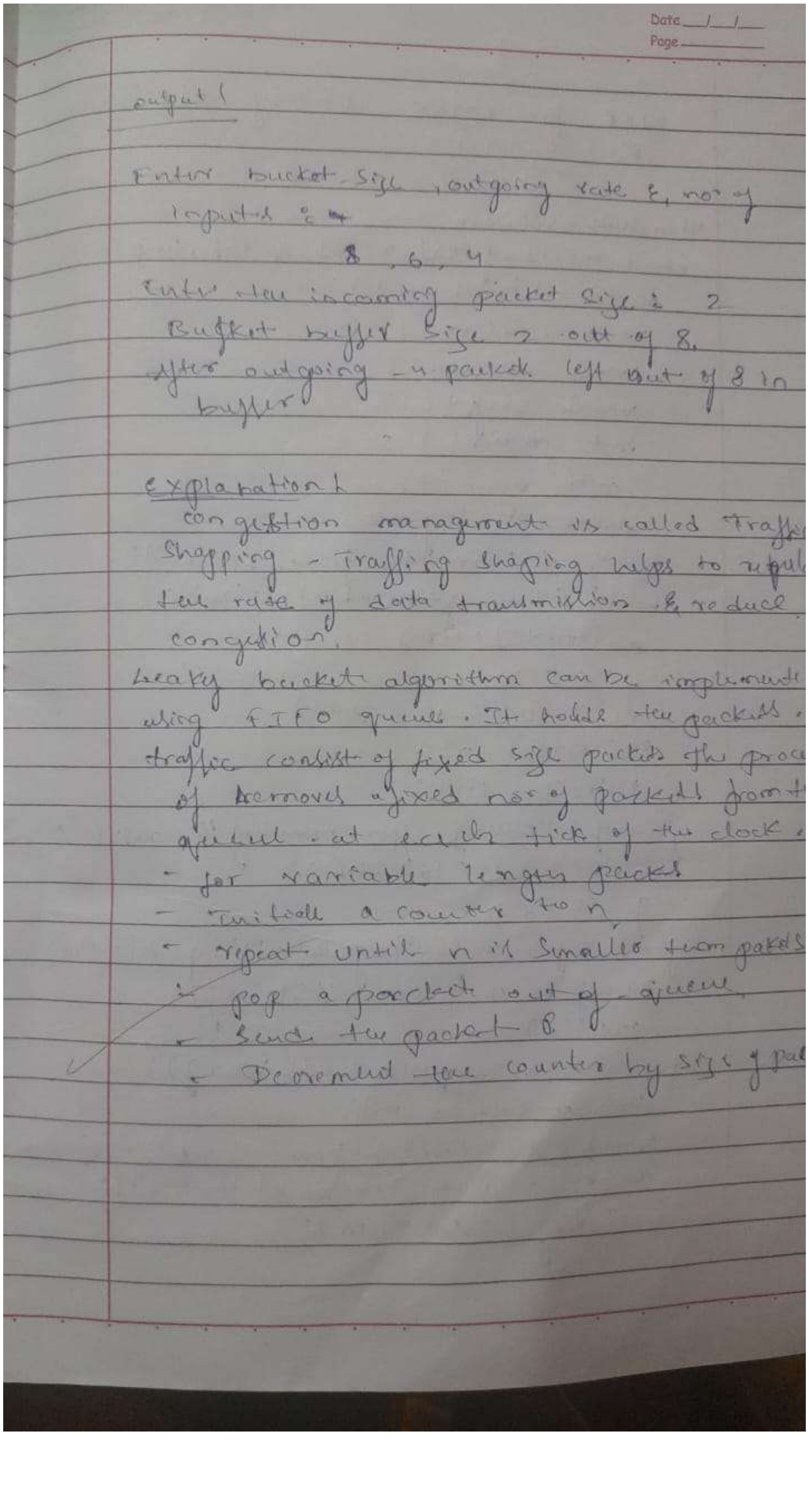
Experiment 14

Write a program for congestion control using Leaky bucket algorithm.



Congestion management –
Leaky Bucket Algorithm :

```
#include < stdio.h >
 int main ( ) {
 int incoming , outgoing , bucket_size , n , store = 0 ;
printf ( "Enter bucket size , outgoing rate & no. of inputs : "
   scanf ( "%d %d %d", &bucket_size , &outgoing , &n
while ( n != 0 ) {
  printf ( "Enter the incoming packet size : " );
  scanf ( "%d", & incoming );
printf ( "Incoming packet size %d \n", incoming );
  if ( incoming <= ( bucket_size - store ) ) {
    Store += incoming ;
 printf ( "Bucket buffer size %d out of %d \n",
   Store , bucket_size );
 } else {
  printf ( "Dropped %d no of packets \n", incoming -
     ( bucket_size - Store ) ) ;
 printf ( "Bucket buffer size %d out of %d \n", Store ,
          bucket_size );
     Store = bucket_size ;
   }
   Store = Store - outgoing ;
printf ( "After outgoing %d packets left out of %d
         in buffer \n", Store , buck size ) ;
         n - - ;
   }
 }
```



Date __/__/__
Page ______

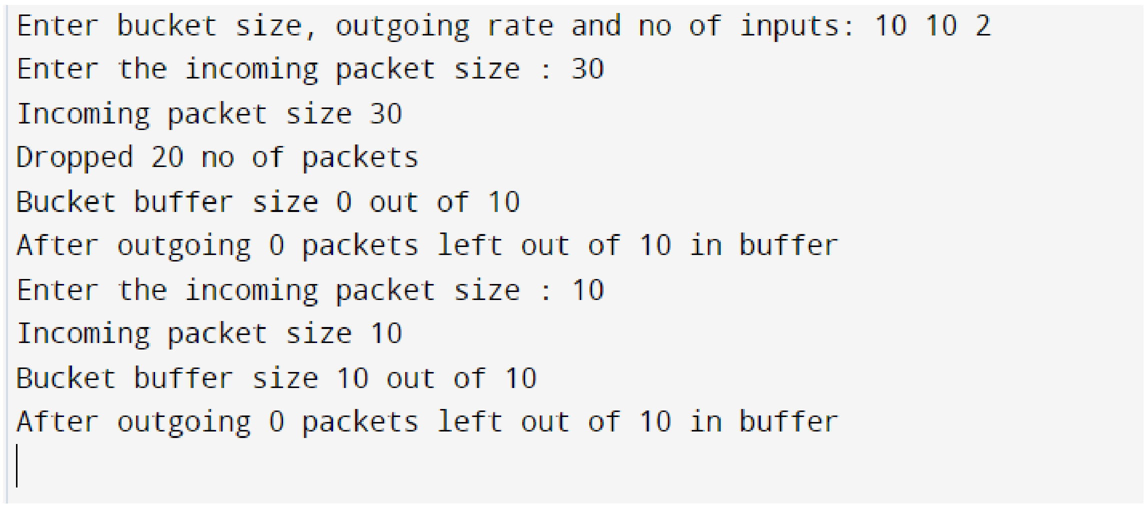
output :

```
Enter bucket size , outgoing rate & no of
  inputs :
              8 , 6 , 4
Enter the incoming packet size : 2
  Bucket buffer size 2 out of 8.
After outgoing -4 packets left out of 8 in
   buffer
```

Explanation :

congestion management is called Traffi
Shapping - Traffing shaping helps to regul
the rate of data transmission & reduce
congestion.
Leaky bucket algorithm can be implemente
using FIFO queue . It holds the packets .
traffic consist of fixed size packets the proce
of removes a fixed no of packets from t
queue at each tick of the clock .
- for variable length packets
- Initialize a counter to n
- repeat until n is smaller than pakets
- pop a packet out of queue
- Send the packet &
- Decrement the counter by size of pa

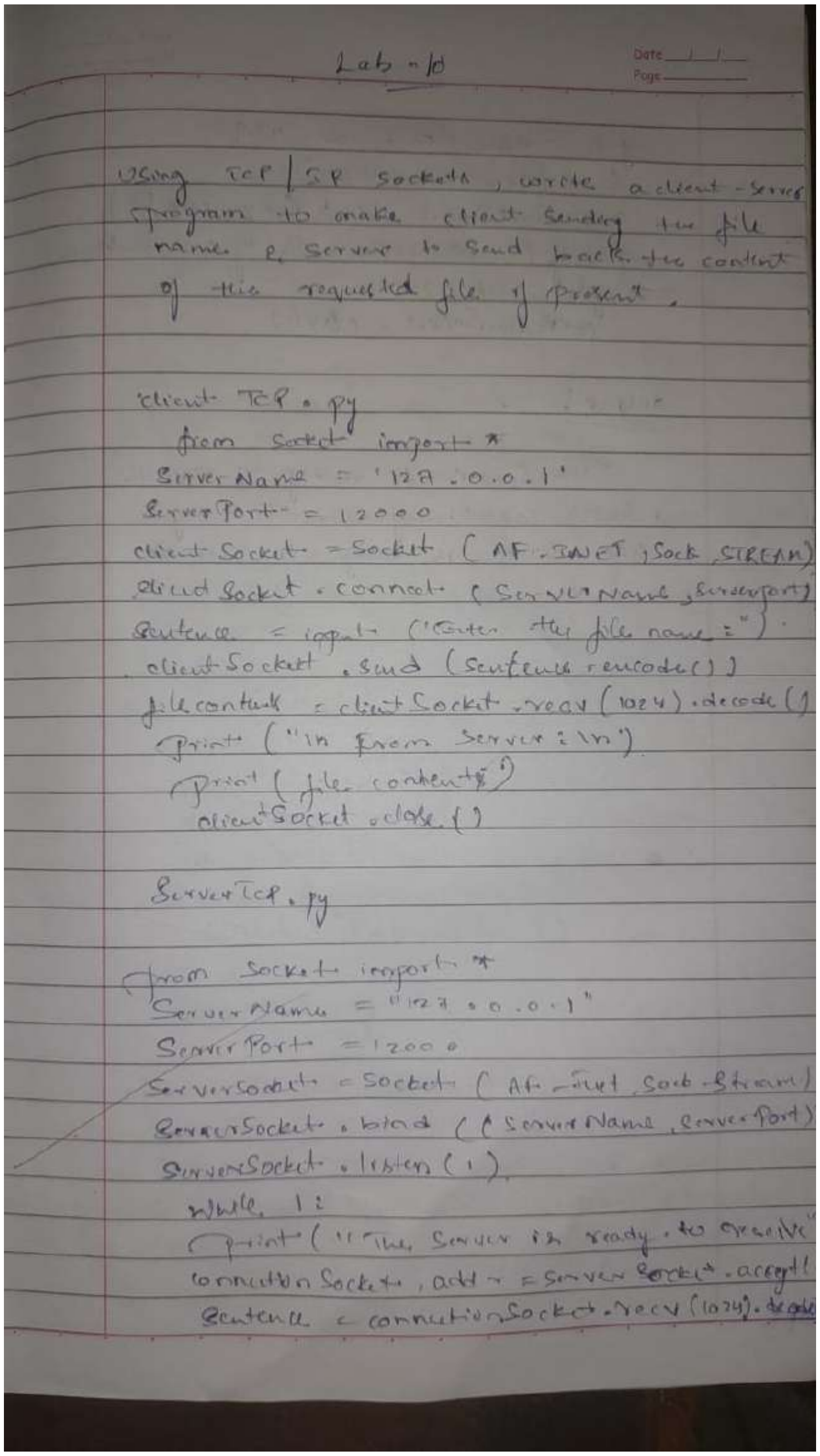
Output:



```
Enter bucket size, outgoing rate and no of inputs: 10 10 2
Enter the incoming packet size : 30
Incoming packet size 30
Dropped 20 no of packets
Bucket buffer size 0 out of 10
After outgoing 0 packets left out of 10 in buffer
Enter the incoming packet size : 10
Incoming packet size 10
Bucket buffer size 10 out of 10
After outgoing 0 packets left out of 10 in buffer
```

Experiment 15

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

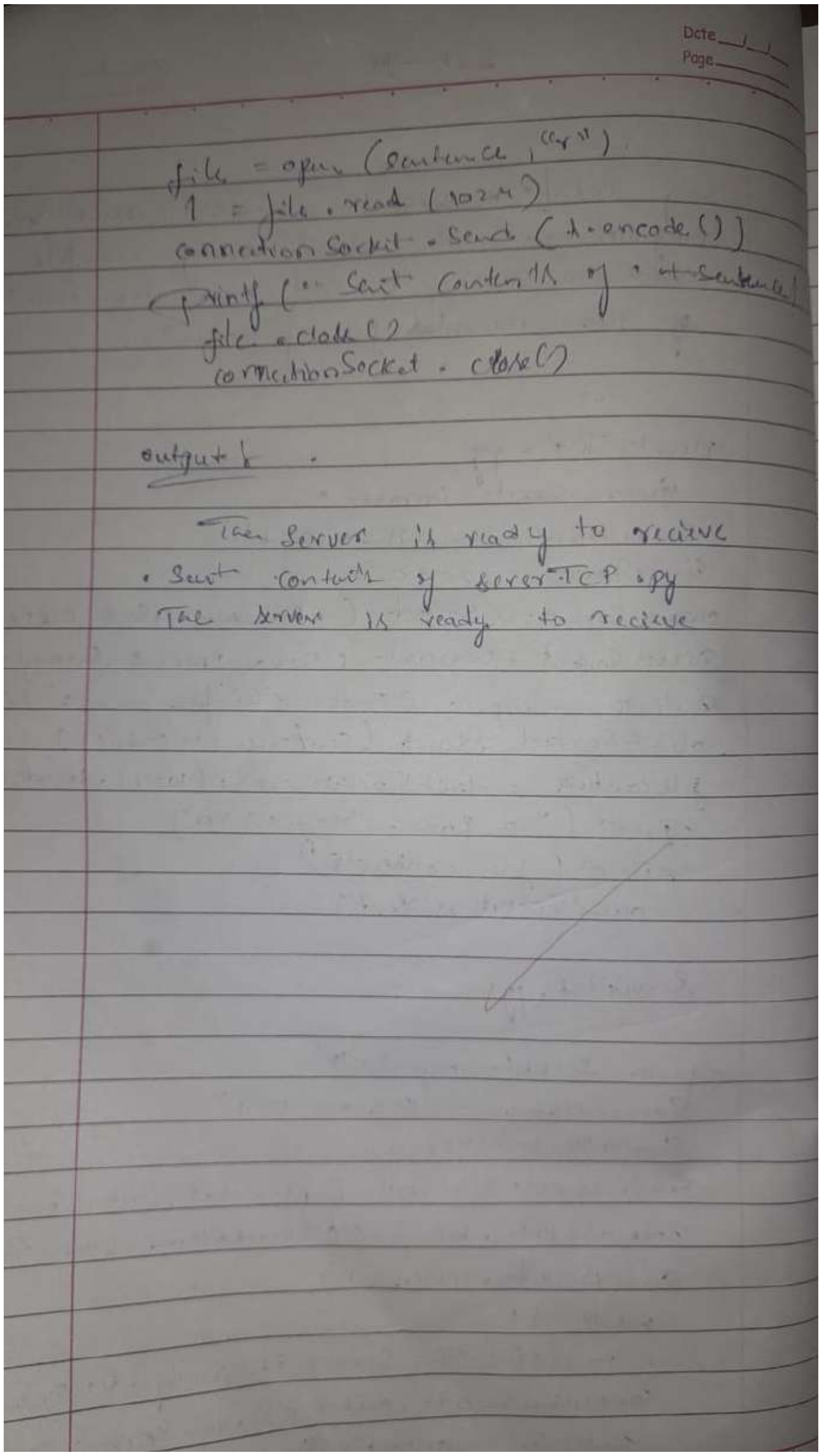


Lab - 10

Date / /
Page

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

```
client TCP . py
  from socket import *
 Server Name = '127 . 0 . 0 . 1'
 Server Port = 12000
client Socket = Socket ( AF . INET , Sock . STREAM)
client Socket . connect ( Server Name , Serverport)
Sentence = input ('Enter the file name :") .
 client Socket . send ( Sentence . encode( ) )
file contents = client Socket . recv ( 1024 ) . decode ( )
 Print ("\n From Server : \n")
  Print ( file contents )
   client Socket . close ( )

Server TCP . py

from Socket import *
 Server Name = "127 . 0 . 0 . 1"
 Server Port = 12000
 Server Socket = Socket ( Af - Inet , Sock - Stream )
 Server Socket . bind ( ( Server Name , Server Port )
 Server Socket . listen ( 1 )
   While 1 :
   Print ( "The Server is ready to receive"
   connection Socket , addr = Server Socket . accept (
   Sentence = connection Socket . recv (1024) . decode
```



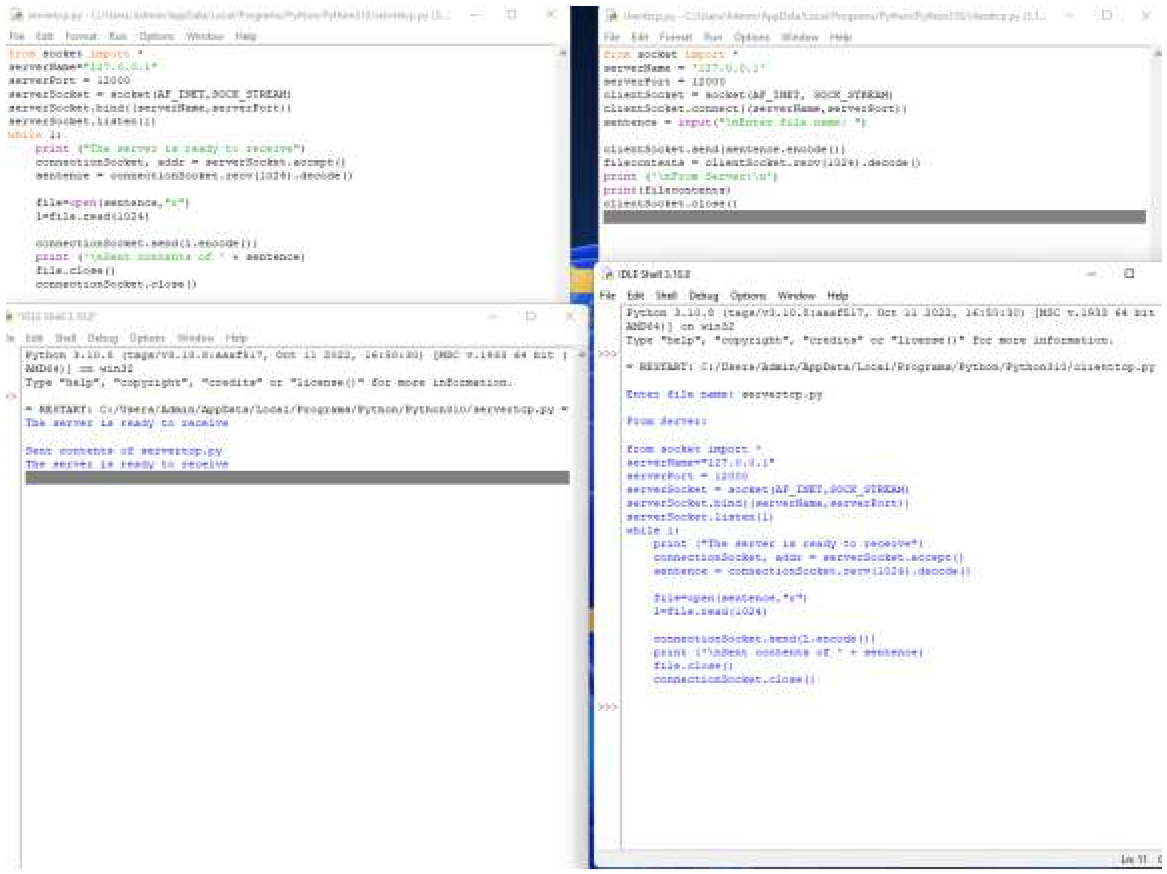
Date / /
Page

```
file = open (sentence , "r")
 l = file . read (1024)
connection Socket . send ( l . encode () )
print ( " Sent contents of " + sentence )
 file . close ()
 connectionSocket . close()
```

output :-

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

Output:



```
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()
```

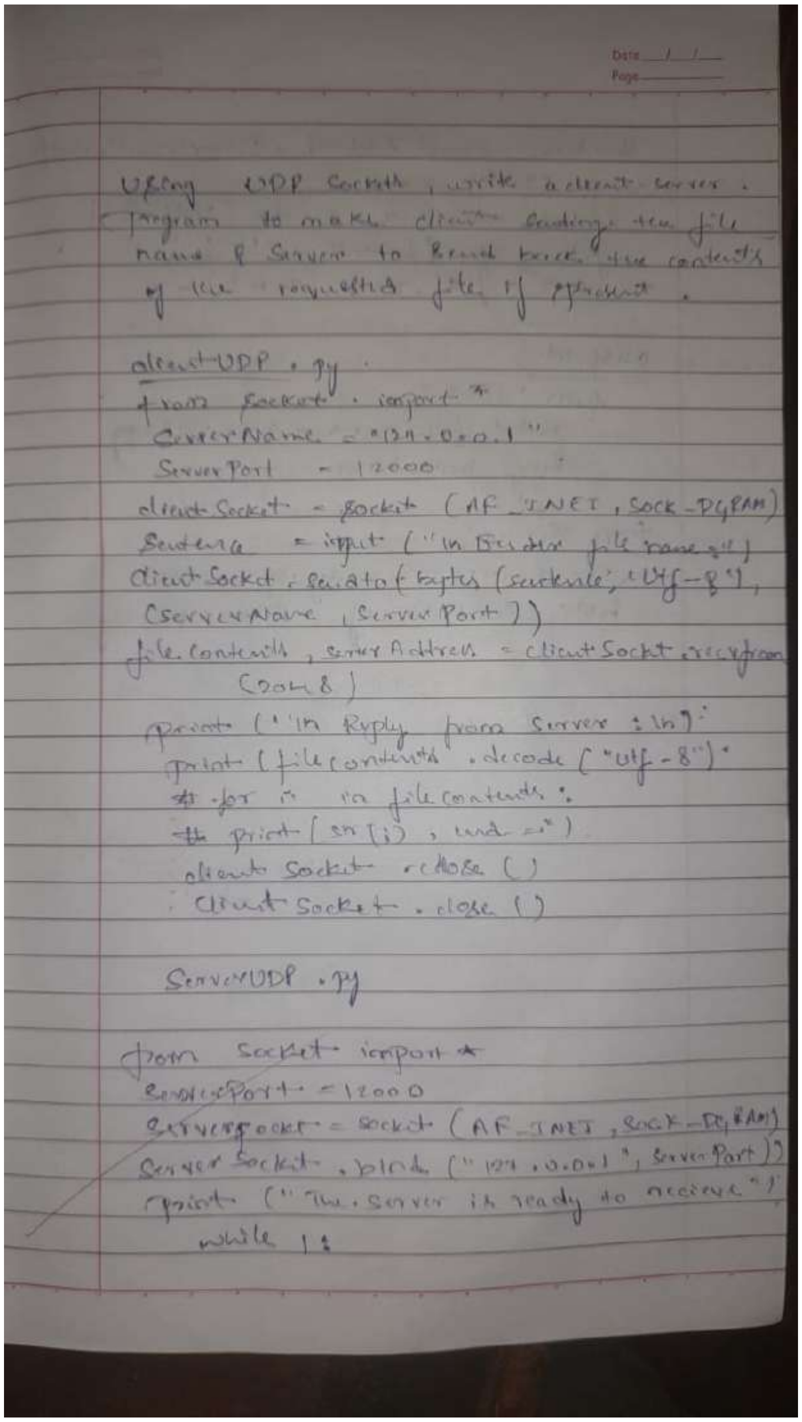
```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/servertcp.py =
The server is ready to receive

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

```
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()
```

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/clienttcp.py

Enter file name: servertcp.py

From Server:

from socket import *
serverName="127.0.0.1"
serverPort = 12000
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1)
while 1:
    print ("The server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    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()

>>>
```

Experiment 16

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



Date / /
Page

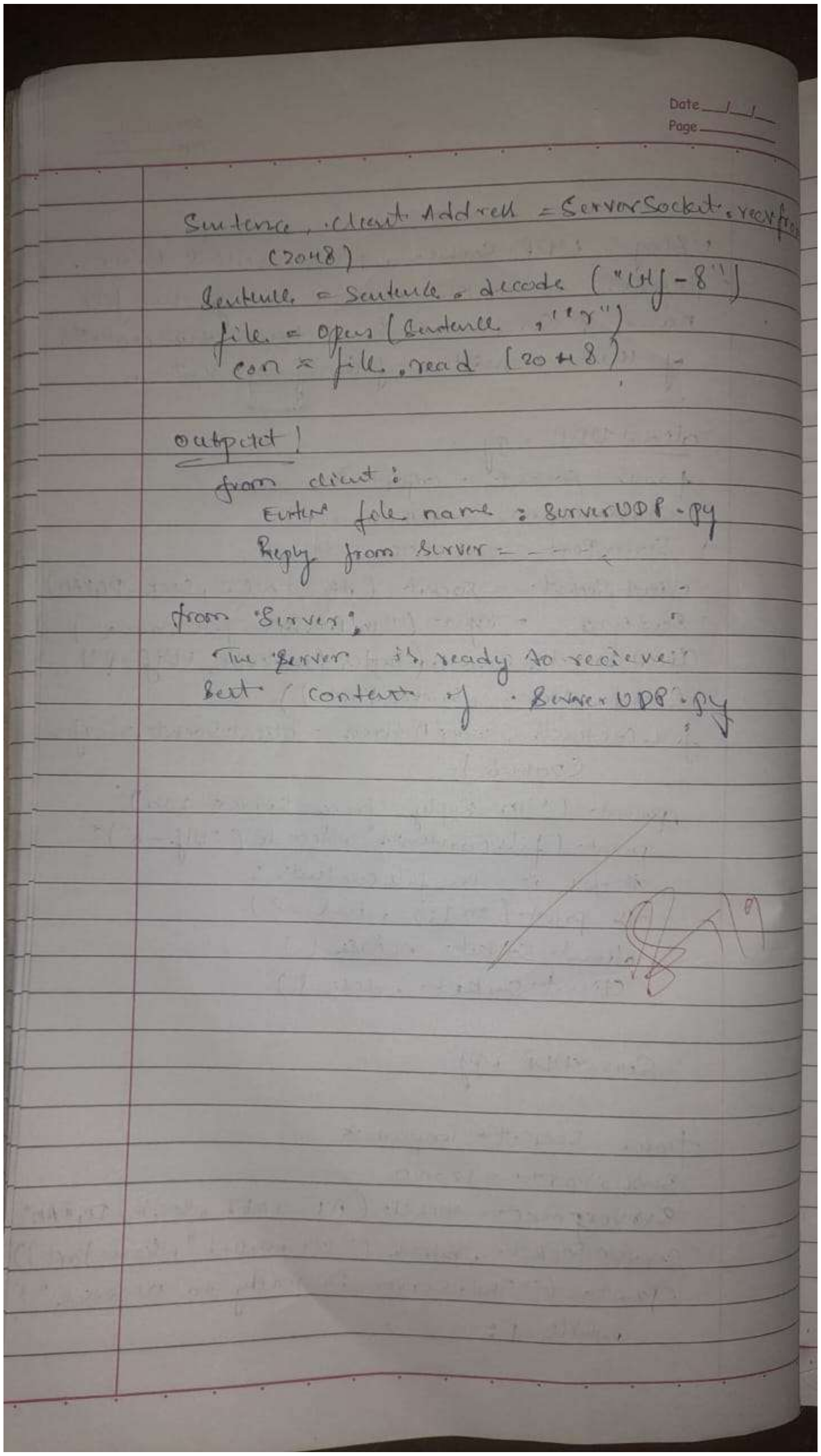
Using UDP sockets, write a client server program to make client sending the file name & server to send back the contents of the requested file if present.

clientUDP . py

```
from socket . import *
 ServerName = "127.0.0.1"
 ServerPort = 12000
clientSocket = socket (AF_INET, SOCK_DGRAM)
Sentence = input ("\n Enter file name :")
clientSocket . sendto ( bytes (sentence, "utf-8"),
 (serverName , ServerPort ))
file contents , serverAddress = clientSocket .recvfrom
        (2048)
 print ('\n Reply from server :\n')
  print (filecontents . decode ("utf-8")
  # for i in file contents :
  # print ( str (i) , end ="")
   client Socket . close ()
   client Socket . close ()
```

ServerUDP .py

```
from socket import *
serverPort = 12000
serverSocket = socket (AF_INET , SOCK_DGRAM)
Server Socket . bind (" 127 .0.0.1 ", ServerPort ))
print (" The server is ready to receive ")
   while 1:
```



Date __/__/__
Page ____

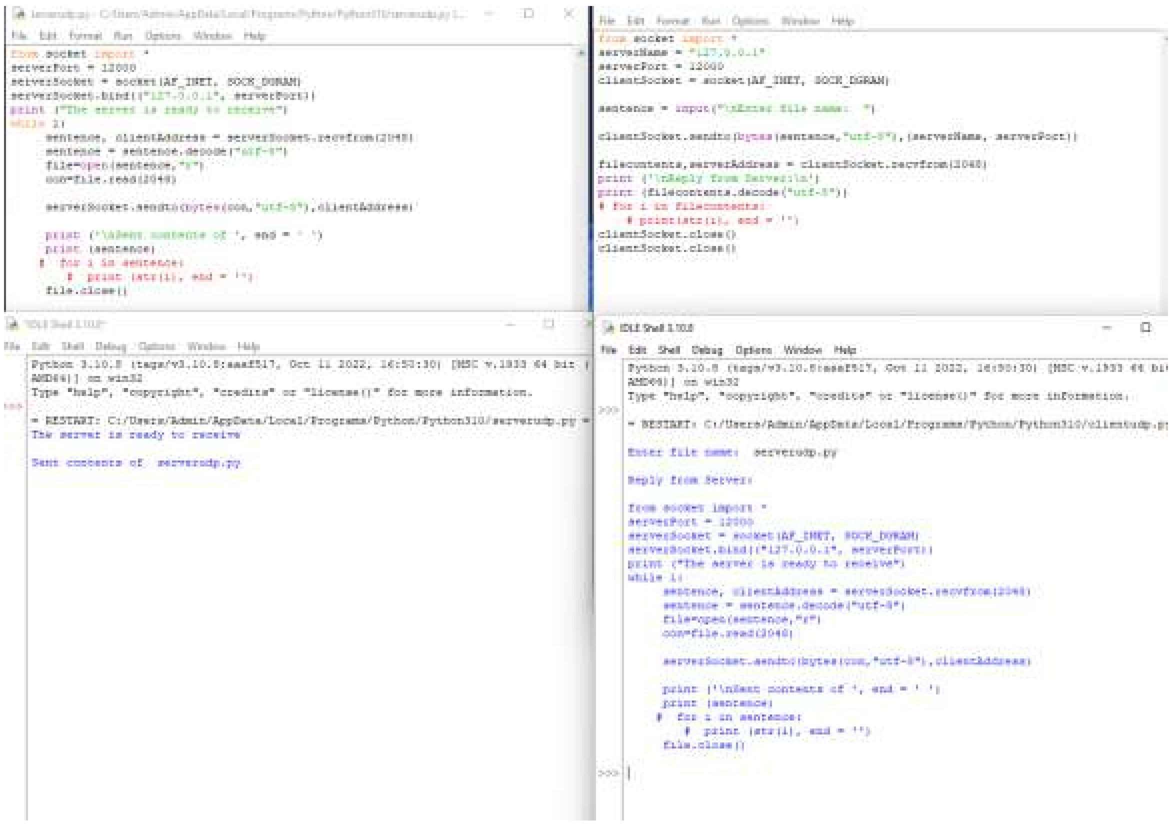
```
Sentence, clientAddress = ServerSocket.recvfro
    (2048)
 Sentence = Sentence.decode ("utf-8")
 file = open (Sentence , "r")
   con = file.read (2048)
```

outputt!

```
  from client :
      Enter file name : serverUDP.py
      Reply from server : - - -

from Server :
  The Server is ready to recieve
  Sent (contents of .ServerUDP.py
```

[illegible]

Output:


File Edit Format Run Options Window Help
IDLE Shell 3.10.8
File Edit Shell Debug Options Window Help

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

     serverSocket.sendto(bytes(con,"utf-8"),clientAddress)

     print ('\nSent contents of ', end = ' ')
     print (sentence)
    #  for i in sentence:
        #  print (str(i), end = '')
     file.close()
```

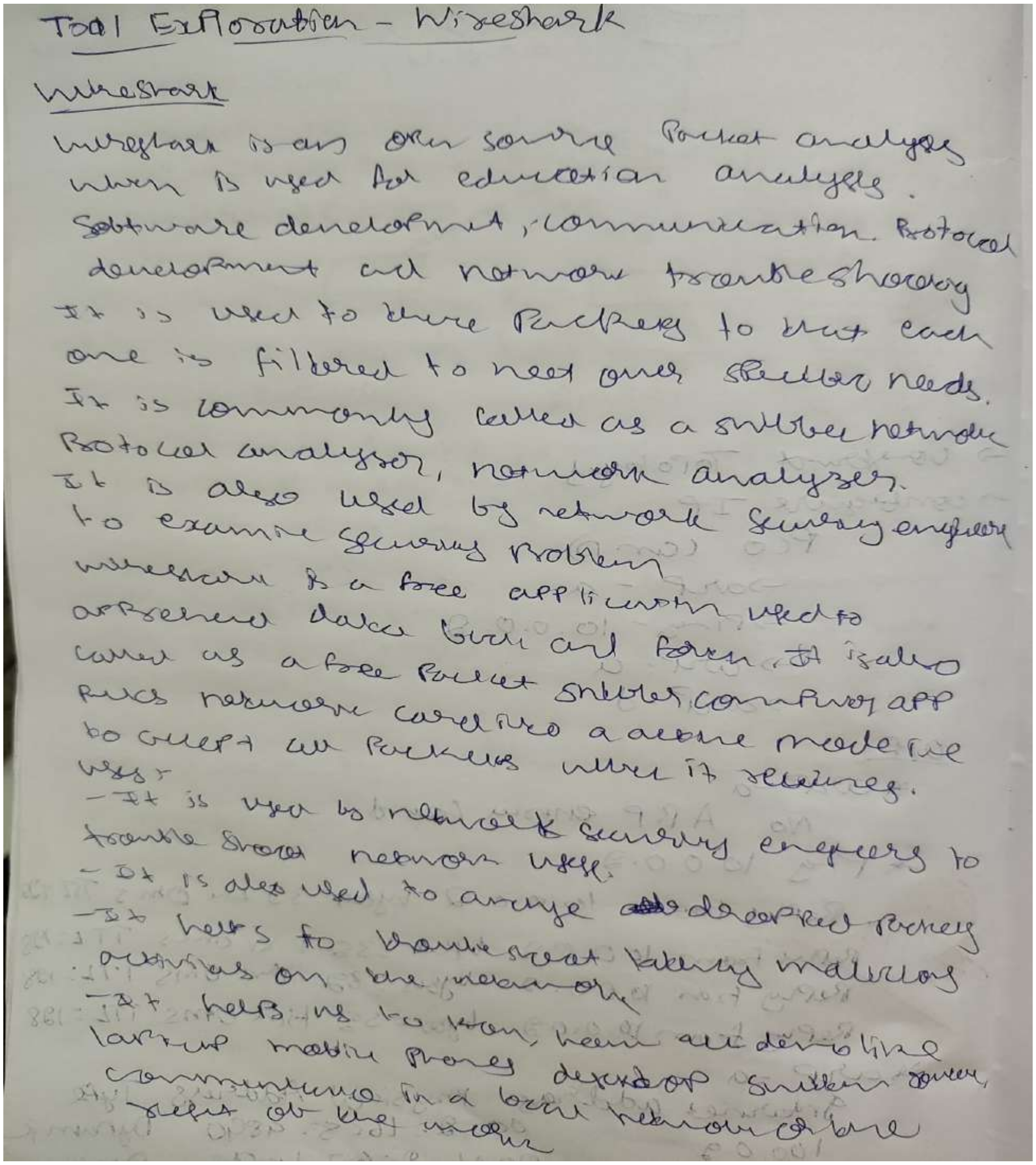
```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)

sentence = input("\nEnter file name:  ")

clientSocket.sendto(bytes(sentence,"utf-8"),(serverName, serverPort))

filecontents,serverAddress = clientSocket.recvfrom(2048)
print ('\nReply from Server:\n')
print (filecontents.decode("utf-8"))
# for i in filecontents:
    # print(str(i), end = '')
clientSocket.close()
clientSocket.close()
```

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/serverudp.py =
The server is ready to receive

Sent contents of  serverudp.py
```

```
Python 3.10.8 (tags/v3.10.8:aaaf517, Oct 11 2022, 16:50:30) [MSC v.1933 64 bi
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python310/clientudp.p

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")
     con=file.read(2048)

     serverSocket.sendto(bytes(con,"utf-8"),clientAddress)

     print ('\nSent contents of ', end = ' ')
     print (sentence)
    #  for i in sentence:
        #  print (str(i), end = '')
     file.close()

>>> |
```

Experiment 17

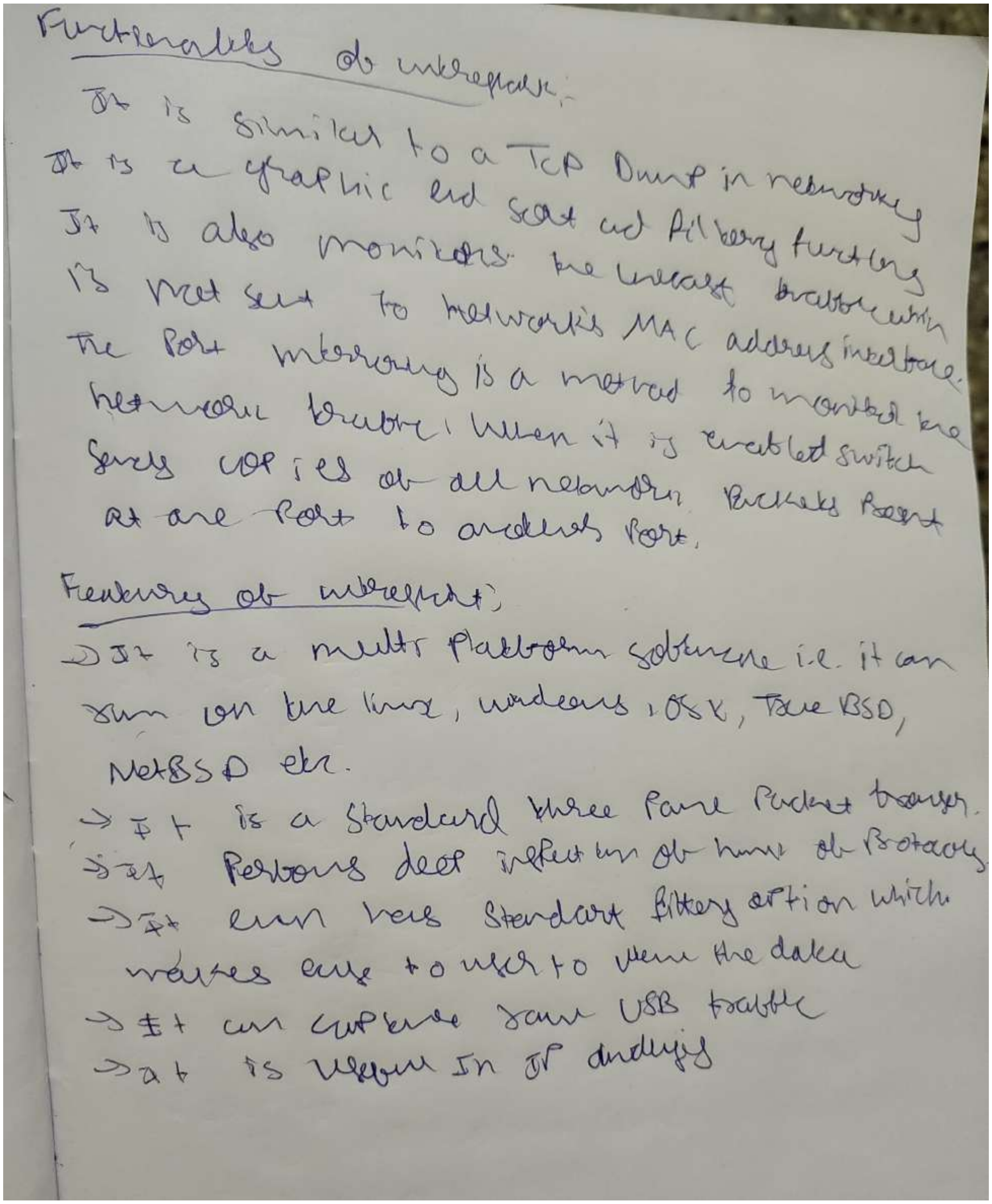
Tool Exploration -Wireshark



Tool Exploration - Wireshark

Wireshark

Wireshark is an open source packet analyser which is used for education analysis. Software development, communication protocol development and network troubleshooting.
It is used to [illegible] packets so that each one is filtered to meet our specific needs.
It is commonly called as a sniffer network protocol analyser, network analyzer.
It is also used by network security engineers to examine security problem.
Wireshark is a free application used to [illegible] data [illegible] and [illegible]. It is also called as a free packet sniffer computer app [illegible] network card into a [illegible] mode to accept all packets which it receives.
- It is used by network security engineers to troubleshoot network issues.
- It is also used to analyze ~~[illegible]~~ dropped packets
- It helps to troubleshoot latency malicious activities on the network.
- It helps us to know, how all devices like laptop mobile phones desktop switches, routers communicate in a local network or the rest of the world



Functionality of wireshark:-

It is similar to a TCP Dump in networking
It is a graphic end sort and filtering functions
It is also monitors the unicast traffic which is not sent to network's MAC address interface.
The Port mirroring is a method to monitor the network traffic. When it is enabled switch sends copies of all network packets present at one port to another port.

Features of wireshark:

-> It is a multi platform software i.e. it can run on the linux, windows, OSX, Free BSD, NetBSD etc.
-> It is a standard three pane packet browser.
-> It performs deep inspection of hundreds of protocols.
-> It even has standard filtering option which makes easy to user to view the data
-> It can capture raw USB traffic
-> It is useful in VoIP analysis