VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

ANIRUDH MULLANGI (1BM20CS016)

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
October-2022 to Feb-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 **ANIRUDH MULLANGI (1BM20CS016),** who is bonafide student of B.M. **S. College of Engineering.**It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of a **Computer Networks- (20CS5PCCON)** work prescribed for the said degree.

Dr. LATHA N.R.
Assistant Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

`

Index

Sl. No.	Date	Experiment Title	Page No.
1		Creating a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.	1
2		Configuring IP address to Routers in Packet Tracer. Exploring the following messages: Ping Responses, Destination unreachable, Request timed out, Reply	4
3		Configuring default route to the Router	7
4		Configuring DHCP within a LAN in a packet Tracer	9
5		Configuring RIP Routing Protocol in Routers	11
6		Demonstration of WEB server and DNS using Packet Tracer	14
7		Write a program for error detecting code using CRC-CCITT (16-bits).	16
8		Write a program for distance vector algorithm to find suitable path for transmission.	20
9		Implement Dijkstra’s algorithm to compute the shortest path for a given topology.	26
10		Write a program for congestion control using Leaky bucket algorithm	29
11		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.	31
12		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.	34

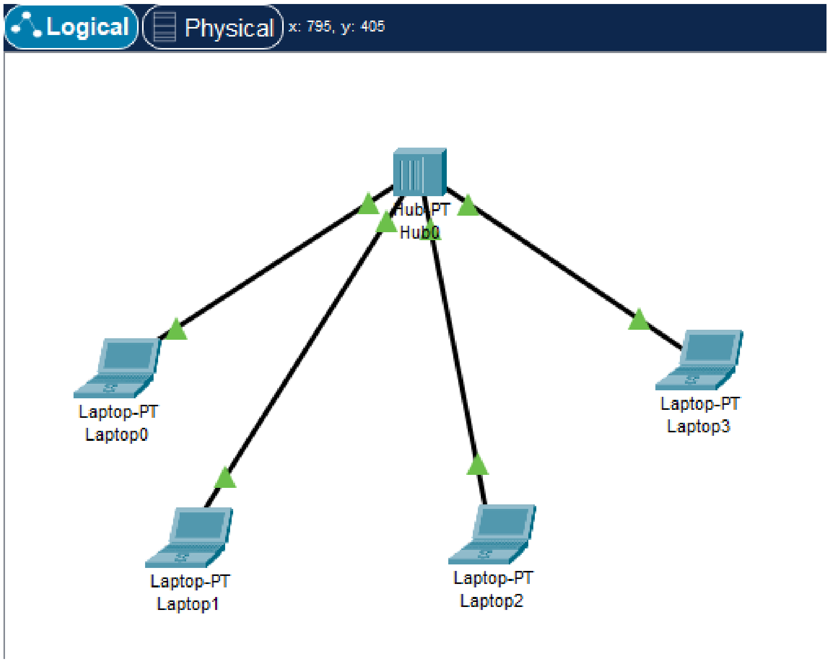
Cycle-1

Experiment No 1

Aim of the program

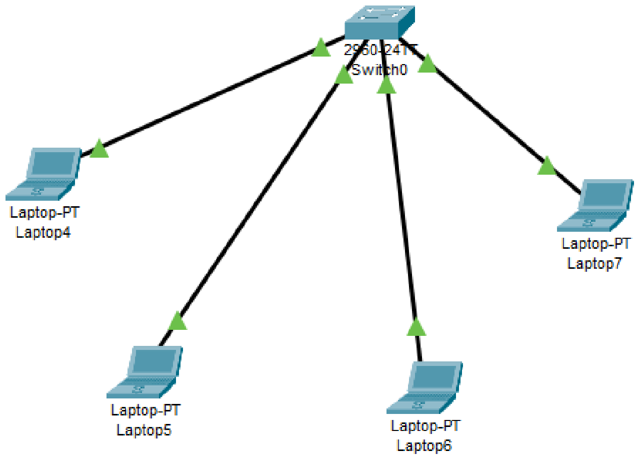
Creating a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.

Hub Topology

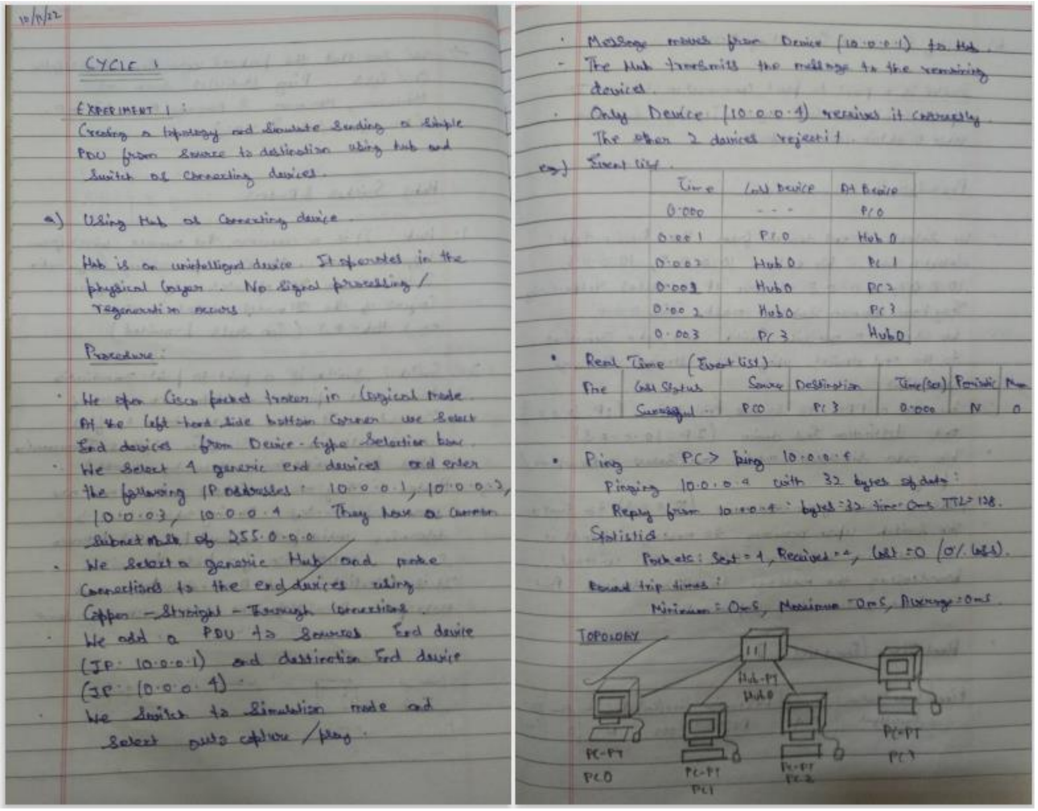


Switch

Topology



Procedure



10/11/22

CYCLE 1

EXPERIMENT 1 :

Creating a topology and Simulate Sending a Simple PDU from Source to destination using hub and Switch as Connecting devices.

a) Using Hub as Connecting device.

Hub is an unintelligent device. It operates in the physical layer. No signal processing / regeneration occurs.

Procedure :

- We open Cisco packet tracer in Logical mode. At the left-hand side bottom Corner we select End devices from Device-type Selection box.
- We Select 4 generic end devices and enter the following IP addresses : 10.0.0.1, 10.0.0.2, 10.0.0.3, 10.0.0.4. They have a Common Subnet mask of 255.0.0.0
- We Select a generic Hub and make Connections to the end devices using Copper - Straight - Through Connections.
- We add a PDU to Sources End device (IP 10.0.0.1) and destination End device (IP 10.0.0.4).
- We Switch to Simulation mode and Select auto capture / play.

- Message moves from Device (10.0.0.1) to Hub.
- The Hub transmits the message to the remaining devices.
- Only Device (10.0.0.4) receives it correctly. The other 2 devices reject it.

eg) Event list.

Time	Last Device	At Device
0.000	- - -	PC0
0.001	PC0	Hub 0
0.002	Hub 0	PC 1
0.002	Hub0	PC2
0.002	Hub0	PC3
0.003	PC3	Hub0

- Real Time (Event list)

Fire	Last Status	Source	Destination	Time(sec)	Periodic	Num
	Successful	PC0	PC3	0.000	N	0

- Ping PC> ping 10.0.0.4

  Pinging 10.0.0.4 with 32 bytes of data :

  Reply from 10.0.0.4 : bytes=32 time=0ms TTL=128.

  Statistics :

  Packets : Sent = 1, Received = 1, Lost = 0 (0% loss).

  Round trip times :

  Minimum = 0mS, Maximum = 0mS, Average = 0mS

TOPOLOGY

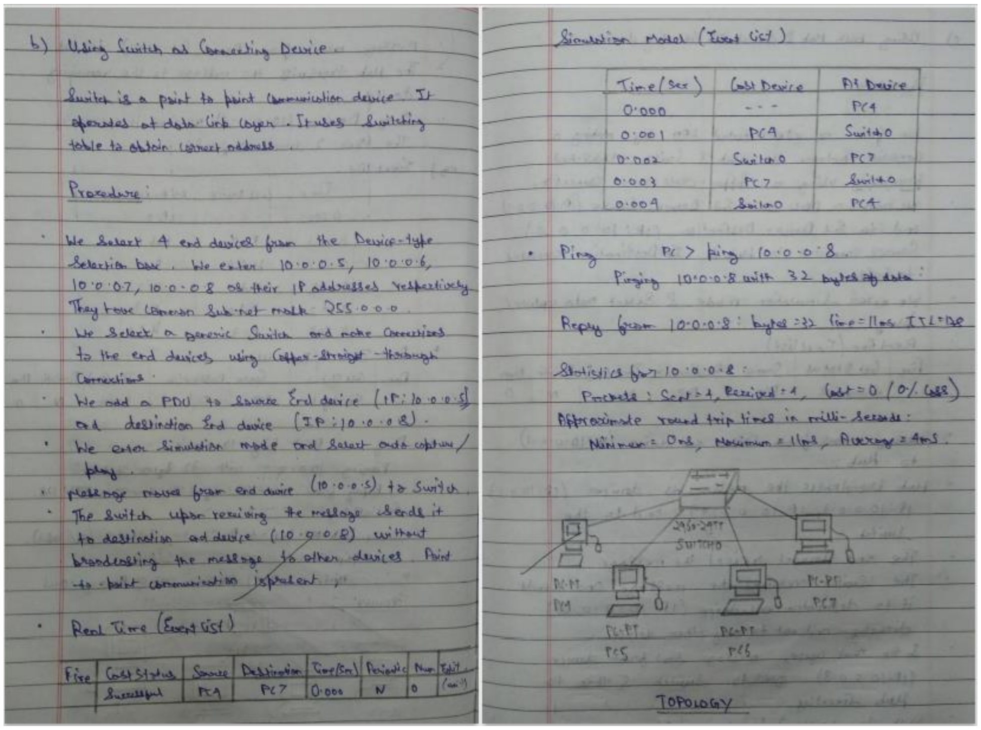
Hub-PT
Hub0

PC-PT
PC0

PC-PT
PC1

PC-PT
PC2

PC-PT
PC3



b) Using Switch as Connecting Device.

Switch is a point to point Communication device. It operates at data link layer. It uses Switching table to obtain correct address.

Procedure :

- We Select 4 end devices from the Device-type Selection box. We enter 10.0.0.5, 10.0.0.6, 10.0.0.7, 10.0.0.8 as their IP addresses respectively. They have Common Sub-net mask 255.0.0.0.
- We Select a generic Switch and make Connections to the end devices using Copper-Straight-through Connections.
- We add a PDU to Source End device (IP: 10.0.0.5) and destination End device (IP : 10.0.0.8).
- We enter Simulation mode and Select auto capture / play.
- Message moves from end device (10.0.0.5) to Switch.
- The Switch upon receiving the message sends it to destination end device (10.0.0.8) without broadcasting the message to other devices. Point-to-point communication is present.

- Real Time (Event List)

Fire	Last Status	Source	Destination	Time(Sec)	Periodic	Num	Edit
	Successful	PC4	PC7	0.000	N	0	(edit)

Simulation Model (Event List)

Time(Sec)	Last Device	At Device
0.000	- - -	PC4
0.001	PC4	Switch0
0.002	Switch0	PC7
0.003	PC7	Switch0
0.004	Switch0	PC4

- Ping PC > ping 10.0.0.8.

  Pinging 10.0.0.8 with 32 bytes of data :

  Reply from 10.0.0.8 : bytes=32 time=11ms TTL=128

  Statistics for 10.0.0.8 :

  Packets : Sent = 1, Received = 1, Lost = 0 (0% Loss)

  Approximate round trip times in milli-seconds :

  Minimum = 0mS, Maximum = 11mS, Average = 4mS.

2960-24TT
SWITCH0

PC-PT
PC4

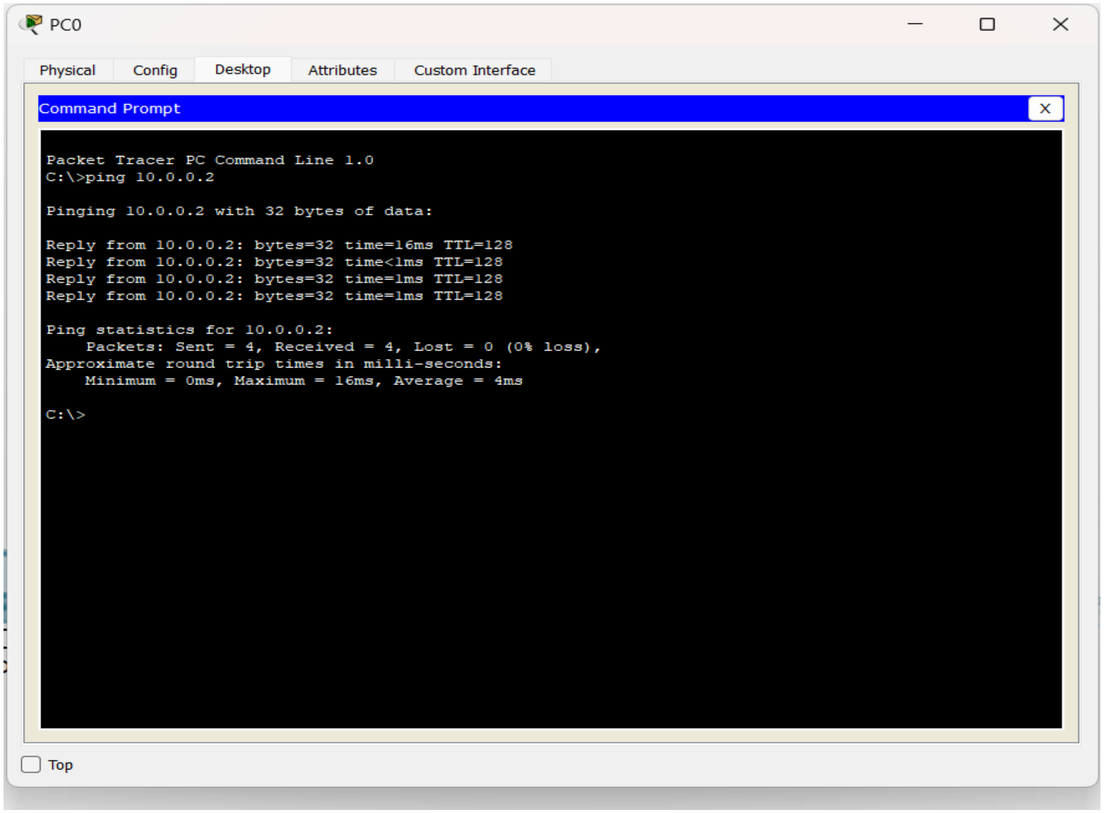
PC-PT
PC5

PC-PT
PC6

PC-PT
PC7

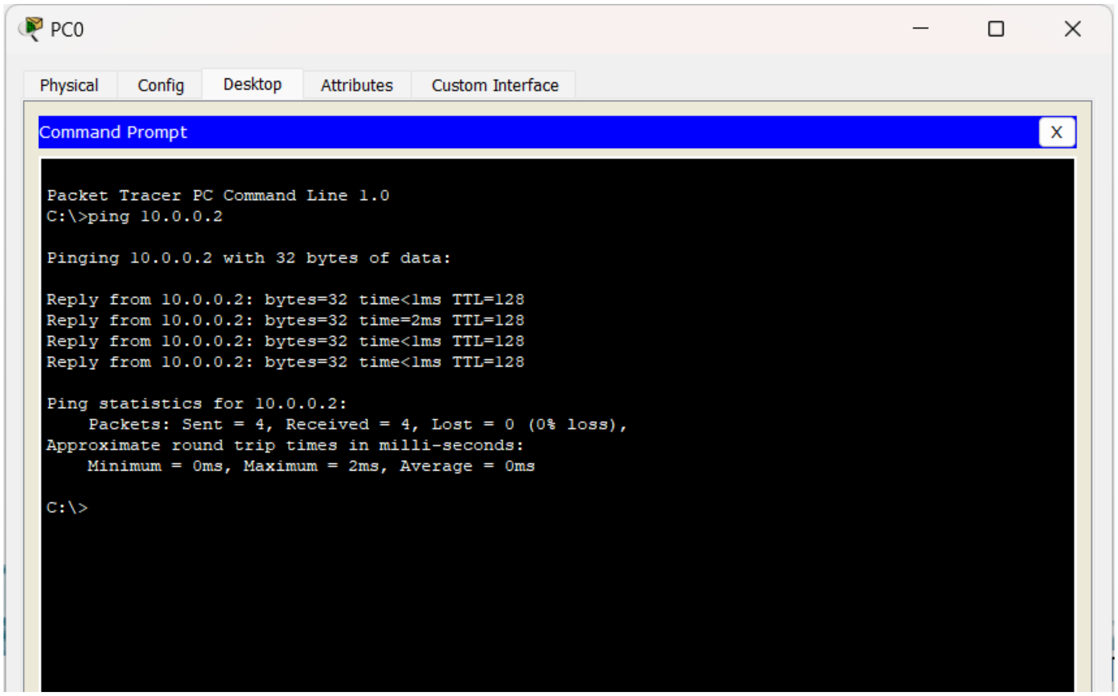
TOPOLOGY

Output



```
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time=16ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time=1ms TTL=128
Reply from 10.0.0.2: bytes=32 time=1ms TTL=128

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 16ms, Average = 4ms

C:\>
```

Output



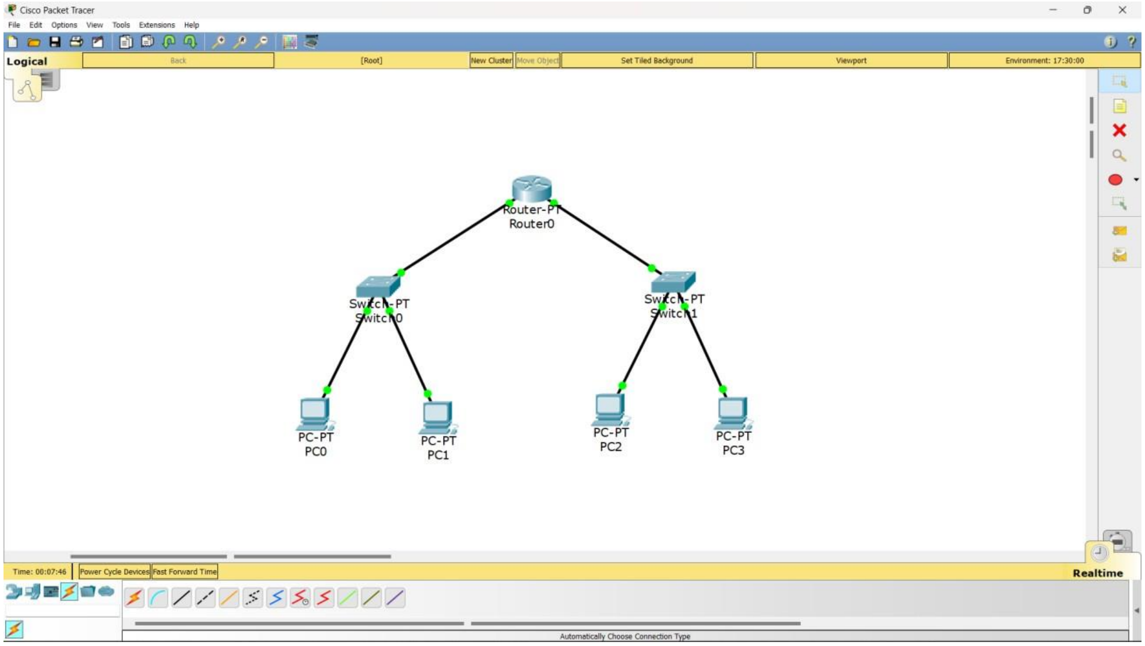
```
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time=2ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 2ms, Average = 0ms

C:\>
```

Experiment No 2

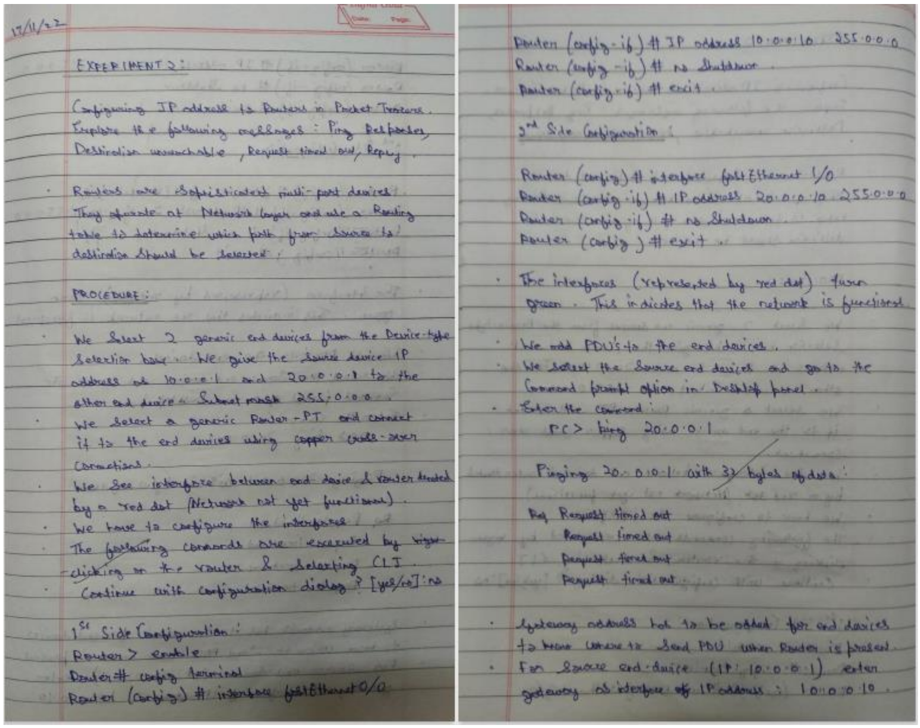
Aim of the program

Configuring IP address to Routers in Packet Tracer. Exploring the following messages: Ping Responses, Destination unreachable, Request timed out, Reply.

Topology



Procedure



17/11/22

EXPERIMENT 2:

Configuring IP address to Routers in Packet Tracer. Explore the following messages : Ping Responses, Destination unreachable, Request timed out, Reply.

- Routers are Sophisticated multi-port devices. They operate at Network layer and use a Routing table to determine which path from Source to destination should be selected.

PROCEDURE:

- We Select 2 generic end devices from the Device-type selection box. We give the source device IP address of 10.0.0.1 and 20.0.0.1 to the other end device. Subnet mask 255.0.0.0.
- We Select a generic Router-PT and connect it to the end devices using copper cross-over connections.
- We see interface between end device & router denoted by a red dot (Network not yet functional).
- We have to configure the interfaces.
- The ~~following~~ commands are executed by right-clicking on the router & Selecting CLI.
- Continue with configuration dialog? [yes/no]: no

1st Side Configuration:

```
Router> enable
Router# config terminal
Router (config) # interface fastEthernet 0/0
Router (config-if) # IP address 10.0.0.10 255.0.0.0
Router (config-if) # no Shutdown.
Router (config-if) # exit.
```

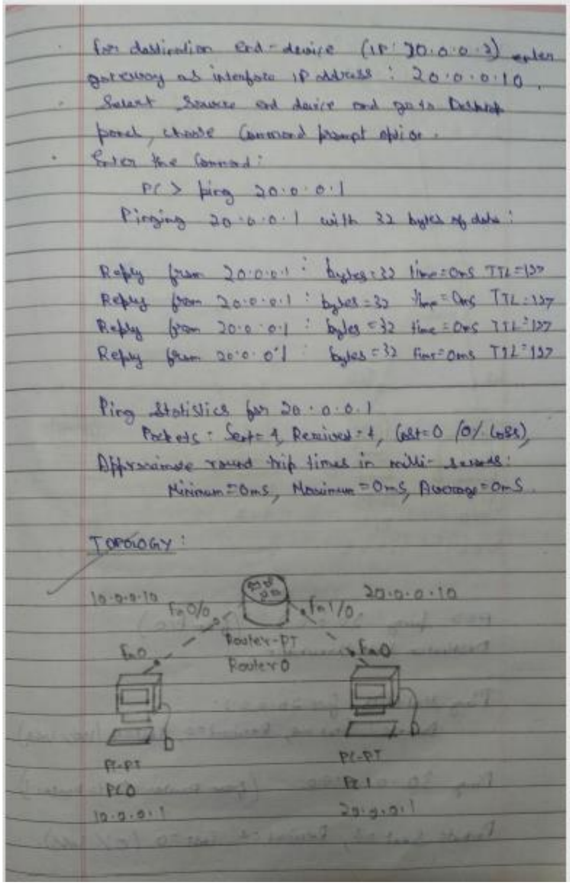
2nd Side Configuration:

```
Router (config) # interface fastEthernet 1/0
Router (config-if) # IP address 20.0.0.10 255.0.0.0
Router (config-if) # no Shutdown.
Router (config) # exit.
```

- The interfaces (represented by red dot) turn green. This indicates that the network is functional.
- We add PDU's to the end devices.
- We select the Source end devices and go to the Command prompt option in Desktop panel.
- Enter the command:

```
PC> ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Req Request timed out
    Request timed out
    Request timed out
    Request timed out
```

- Gateway address has to be added for end devices to know where to Send PDU when Router is present.
- For Source end device (IP: 10.0.0.1) enter gateway as interface of IP address : 10.0.0.10.

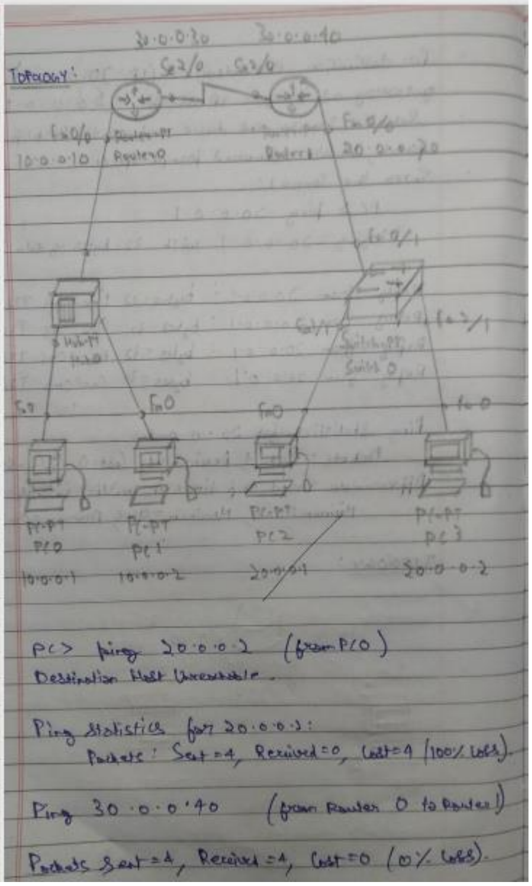


- for destination end-device (IP: 20.0.0.2) enter gateway as interface IP address : 20.0.0.10.
- Select Source end device and go to Desktop panel, choose Command prompt option.
- Enter the Command:

```
PC> ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:

Reply from 20.0.0.1 : bytes=32 time=0ms TTL=127
Reply from 20.0.0.1 : bytes=32 time=0ms TTL=127
Reply from 20.0.0.1 : bytes=32 time=0ms TTL=127
Reply from 20.0.0.1 : bytes=32 time=0ms TTL=127

Ping statistics for 20.0.0.1
    Packets : Sent=4, Received=4, Lost=0 (0% Loss),
Approximate round trip times in milli-seconds:
    Minimum=0ms, Maximum=0ms, Average=0ms.
```

TOPOLOGY :

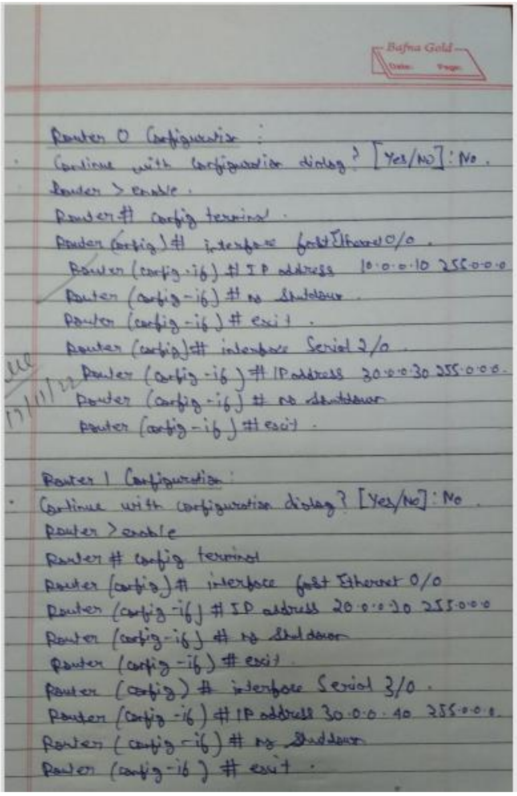
10.0.0.10
Fa0/0
Fa1/0
20.0.0.10
Router-PT
Router0
Fa0
Fa0
PC-PT
PC0
10.0.0.1
PC-PT
PC1
20.0.0.1



TOPOLOGY :

30.0.0.30
30.0.0.40
Se2/0
Se3/0
Fa0/0
Router-PT
10.0.0.10
Router0
Router1
Fa0/0
20.0.0.20
Fa0/1
Hub-PT
Hub0
Fa1/1
Fa2/1
Switch-PT
Switch0
Fa0
Fa0
Fa0
Fa0
PC-PT
PC0
10.0.0.1
PC-PT
PC1
10.0.0.2
PC-PT
PC2
20.0.0.1
PC-PT
PC3
20.0.0.2

```
PC> ping 20.0.0.2 (from PC0)
Destination Host Unreachable.

Ping statistics for 20.0.0.2:
    Packets : Sent=4, Received=0, Lost=4 (100% Loss).

Ping 30.0.0.40 (from Router 0 to Router1)

Packets Sent=4, Received=4, Lost=0 (0% Loss).
```



Router 0 Configuration :

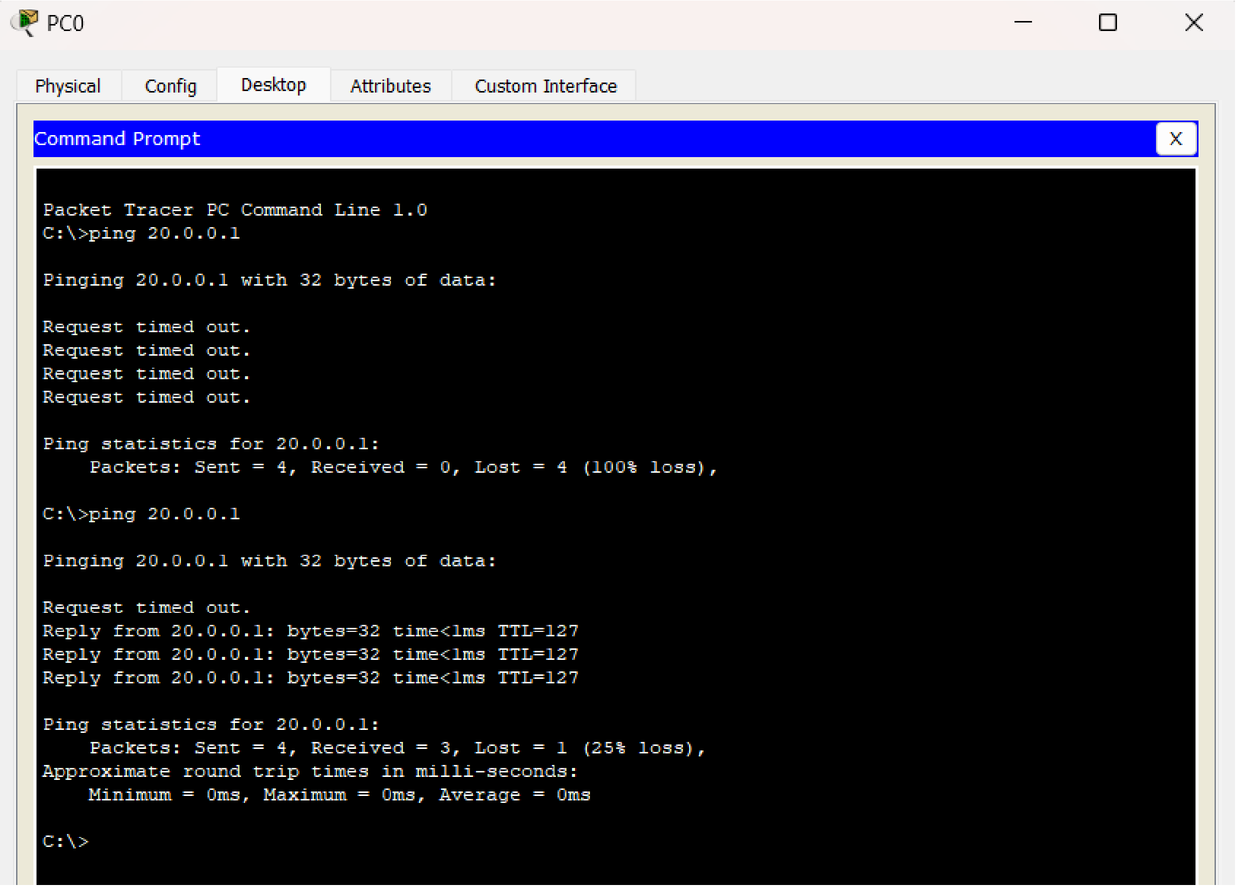
```
Continue with configuration dialog? [Yes/No]: No.
Router > enable.
Router# config terminal.
Router(config)# interface fastEthernet0/0.
 Router (config-if) # IP address 10.0.0.10 255.0.0.0
 Router (config-if) # no Shutdown.
 Router (config-if) # exit .
 Router (config)# interface Serial 2/0.
  Router (config-if) # IP address 30.0.0.30 255.0.0.0.
  Router (config-if) # no shutdown
  Router (config-if) # exit .
```

17/11/22

Router 1 Configuration :

```
Continue with configuration dialog? [Yes/No]: No .
Router > enable
Router # config terminal
Router (config)# interface fast Ethernet 0/0
Router (config-if) # IP address 20.0.0.20 255.0.0.0
Router (config-if) # no Shutdown
Router (config-if) # exit.
Router (config) # interface Serial 3/0 .
Router (config-if) # IP address 30.0.0.40 255.0.0.0.
Router (config-if) # no Shutdown
Router (config-if) # exit .
```

Output



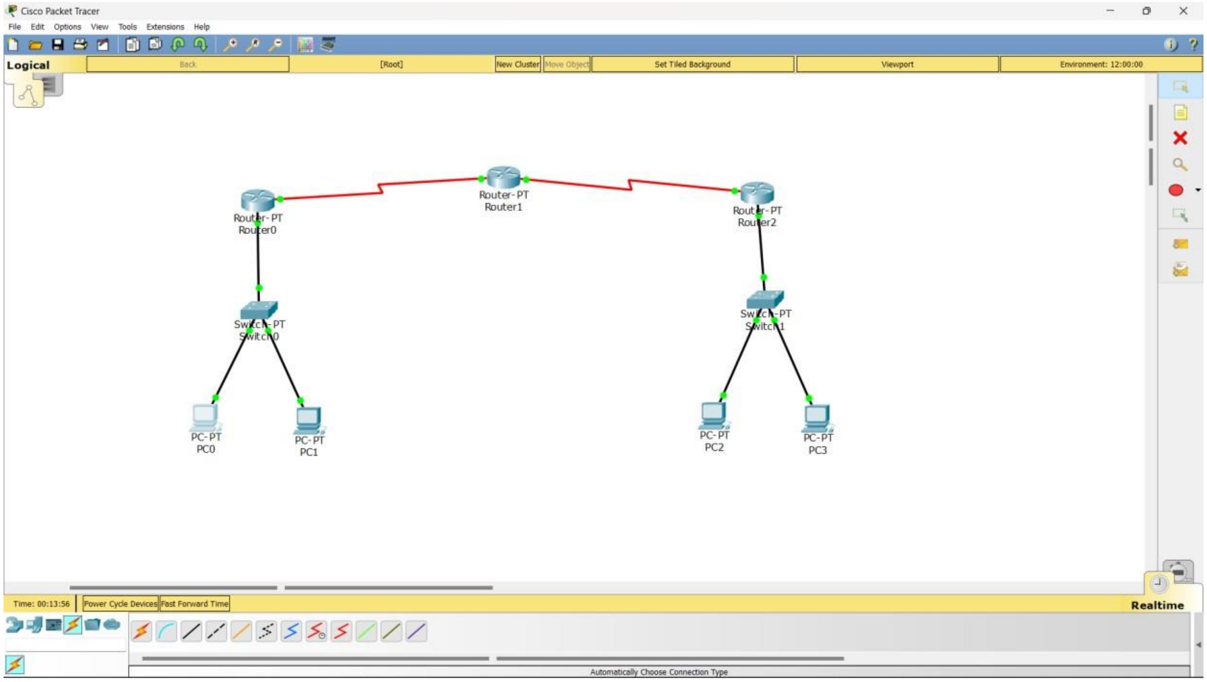
```
Packet Tracer PC Command Line 1.0
C:\>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

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

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

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<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127

Ping statistics for 20.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\>
```

Experiment No 3

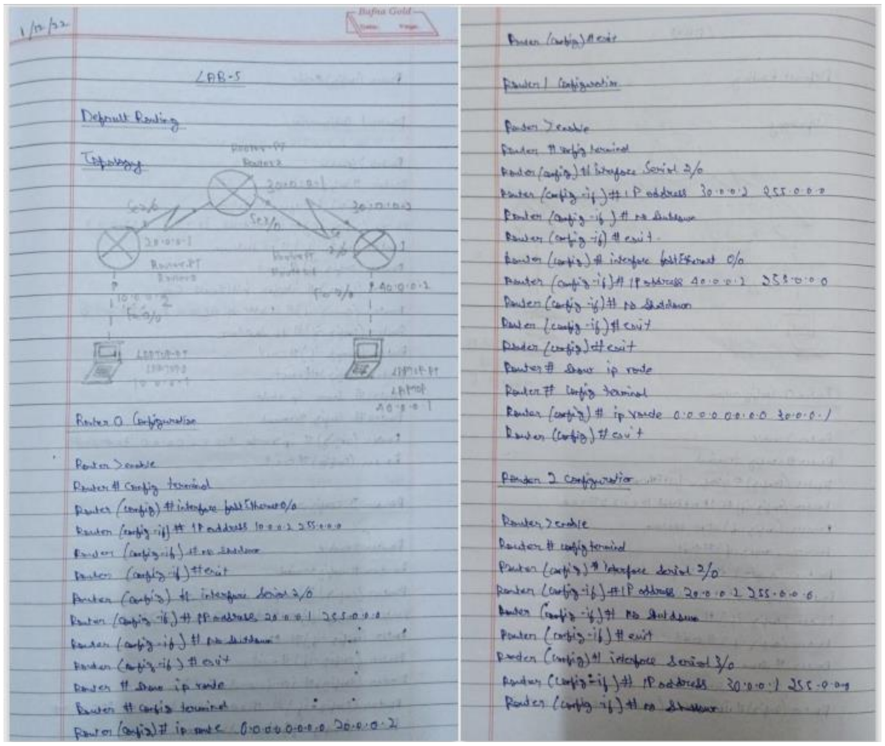
Aim of the program

Configuring default route to the Router

Topology



Procedure


Bafna Gold
Router-PT
Router1
20.0.0.1
30.0.0.2
Se2/0
Se3/0
20.0.0.1
Router-PT
Router0
Router-PT
Router2
2/0
40.0.0.2
10.0.0.2
Fa0/0
Fa0/0
Laptop-PT
Laptop0
10.0.0.1
Laptop-PT
Laptop1
40.0.0.1

1/12/22

LAB-5

Default Routing

Topology

Router 0 Configuration

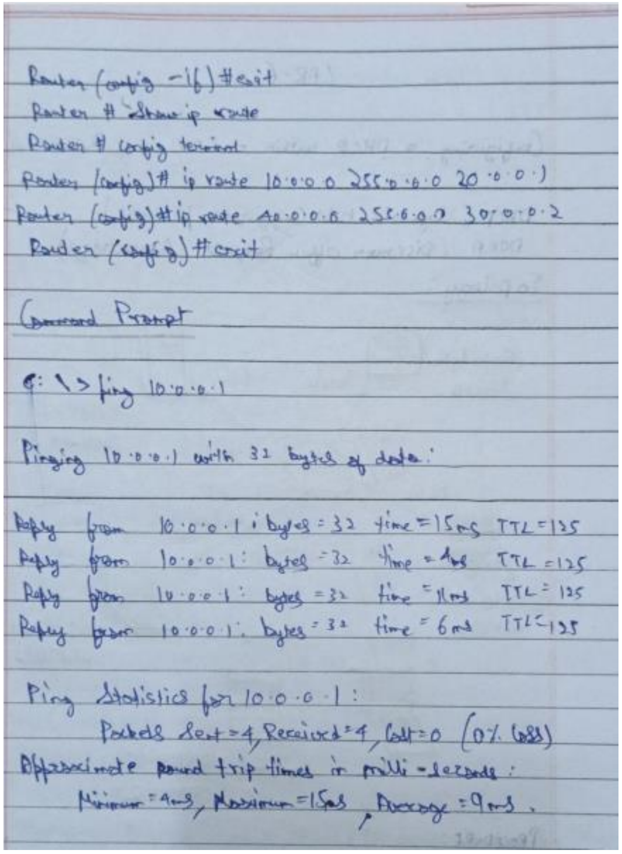
```
Router>enable
Router# Config terminal
Router (config) # interface fastEthernet0/0
Router (config-if) # IP address 10.0.0.2 255.0.0.0
Router (config-if) # no shutdown
Router (config-if) # exit
Router (config) # interface Serial 2/0
Router (config-if) # IP address 20.0.0.1 255.0.0.0
Router (config-if) # no shutdown
Router (config-if) # exit
Router # Show ip route
Router # config terminal
Router (config) # ip route 0.0.0.0 0.0.0.0 20.0.0.2
Router (config) # exit
```

Router 1 Configuration

```
Router>enable
Router # config terminal
Router (config) # interface Serial 2/0
Router (config-if) # IP address 30.0.0.2 255.0.0.0
Router (config-if) # no shutdown
Router (config-if) # exit
Router (config) # interface fastEthernet 0/0
Router (config-if) # IP address 40.0.0.2 255.0.0.0
Router (config-if) # no shutdown
Router (config-if) # exit
Router (config) # exit
Router # Show ip route
Router # Config terminal
Router (config) # ip route 0.0.0.0 0.0.0.0 30.0.0.1
Router (config) # exit
```

Router 2 Configuration

```
Router>enable
Router # config terminal
Router (config) # interface serial 2/0
Router (config-if) # IP address 20.0.0.2 255.0.0.0
Router (config-if) # no shutdown
Router (config-if) # exit
Router (config) # interface Serial 3/0
Router (config-if) # IP address 30.0.0.1 255.0.0.0
Router (config-if) # no Shutdown
```

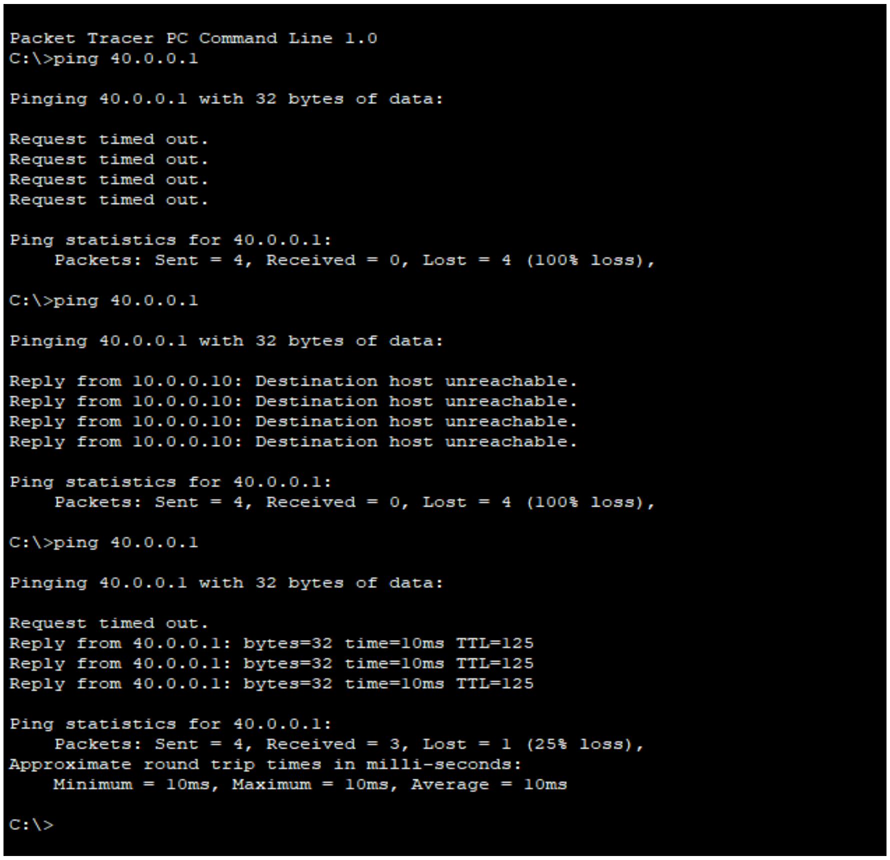


```
Router (config -if)#exit
Router # show ip route
Router # config terminal
Router (config)# ip route 10.0.0.0 255.0.0.0 20.0.0.1
Router (config)#ip route 40.0.0.0 255.0.0.0 30.0.0.2
Router (config)#exit
```

Command Prompt

```
C:\>ping 10.0.0.1

Pinging 10.0.0.1 with 32 bytes of data:

Reply from 10.0.0.1: bytes=32 time=15ms TTL=125
Reply from 10.0.0.1: bytes=32 time=4ms TTL=125
Reply from 10.0.0.1: bytes=32 time=11ms TTL=125
Reply from 10.0.0.1: bytes=32 time=6ms TTL=125

Ping Statistics for 10.0.0.1:
        Packets Sent=4, Received=4, Lost=0 (0% loss)
Approximate round trip times in milli-seconds:
     Minimum=4ms, Maximum=15ms, Average=9ms.
```

Output



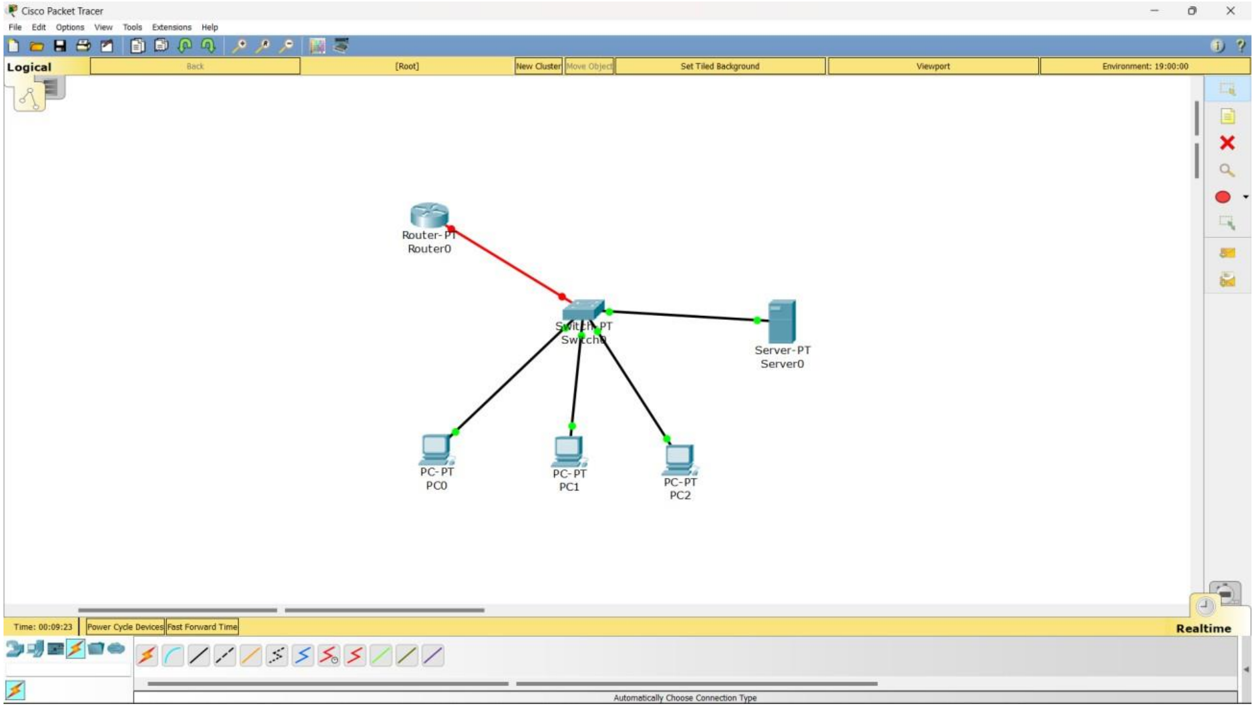
```
Packet Tracer PC Command Line 1.0
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

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=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125
Reply from 40.0.0.1: bytes=32 time=10ms TTL=125

Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
    Minimum = 10ms, Maximum = 10ms, Average = 10ms

C:\>
```

Experiment No 4

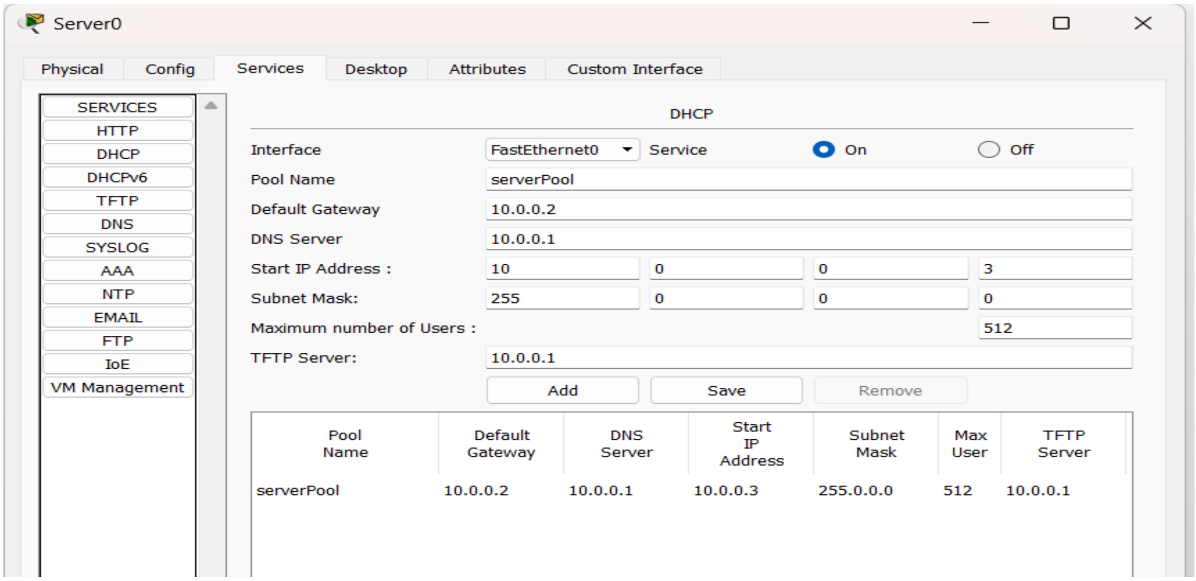
Aim of the program

Configuring DHCP within a LAN in a packet Tracer

Topology



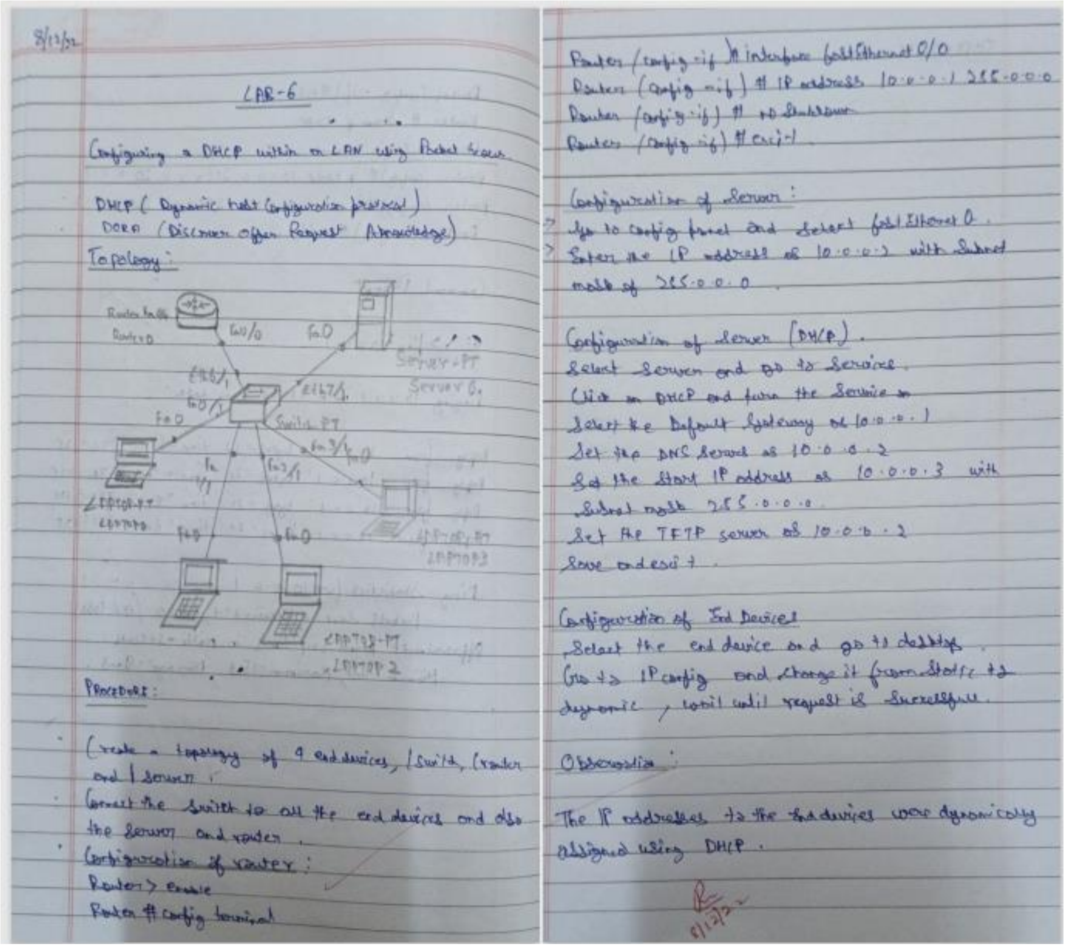
Procedure



DHCP

Interface	FastEthernet0	Service	● On	○ Off
Pool Name	serverPool			
Default Gateway	10.0.0.2			
DNS Server	10.0.0.1			
Start IP Address :	10	0	0	3
Subnet Mask:	255	0	0	0
Maximum number of Users :				512
TFTP Server:	10.0.0.1			

Pool Name	Default Gateway	DNS Server	Start IP Address	Subnet Mask	Max User	TFTP Server
serverPool	10.0.0.2	10.0.0.1	10.0.0.3	255.0.0.0	512	10.0.0.1



8/12/22

LAB-6

Configuring a DHCP within a LAN using Packet Tracer

DHCP (Dynamic Host Configuration protocol)
DORA (Discover Offer Request Acknowledge)

Topology:

Router PT
Router0
Gi0/0
Fa0
Eth6/1
Eth7/1
Fa0/1
Fa0
Switch-PT
Server-PT
Server0
Fa3/1
Fa0
Fa1/1
Fa2/1
Fa0
Fa0
Laptop-PT
Laptop0
Laptop-PT
Laptop3
Laptop-PT
Laptop2

PROCEDURE:

- Create a topology of 4 end devices, 1 Switch, 1 router and 1 server.
- Connect the Switch to all the end devices and also the Server and router.
- Configuration of router:

Router> enable
Router # config terminal
Router (config-if)# interface fastEthernet 0/0
Router (config-if)# IP address 10.0.0.1 255.0.0.0
Router (config-if)# no Shutdown
Router (config-if)# exit.

Configuration of Server:

> Go to config panel and Select fastEthernet 0.
> Enter the IP address as 10.0.0.2 with Subnet mask of 255.0.0.0.

Configuration of Server (DHCP).

Select Server and go to Services.
Click on DHCP and turn the Service on
Select the Default Gateway as 10.0.0.1
Set the DNS Server as 10.0.0.2
Set the Start IP address as 10.0.0.3 with Subnet mask 255.0.0.0.
Set the TFTP server as 10.0.0.2
Save and exit.

Configuration of End Devices

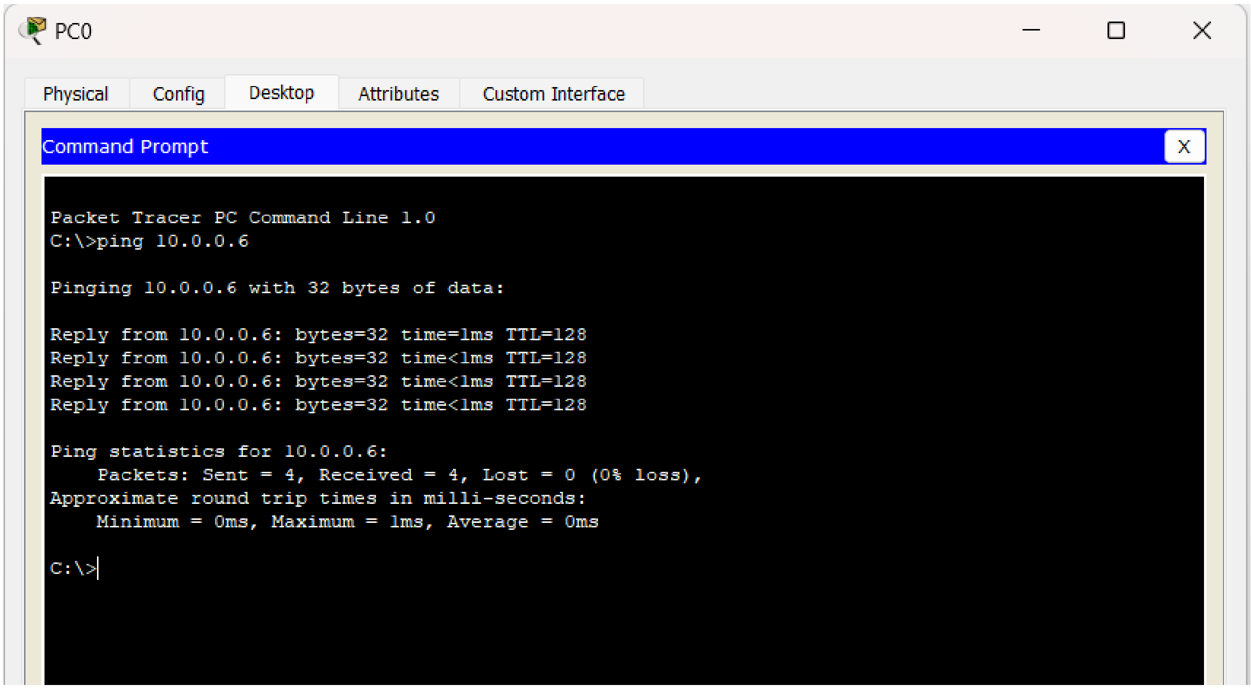
Select the end device and go to desktop.
Go to IP config and change it from Static to dynamic, wait until request is Successful.

Observation:

The IP addresses to the end devices were dynamically assigned using DHCP.

8/12/22

Output



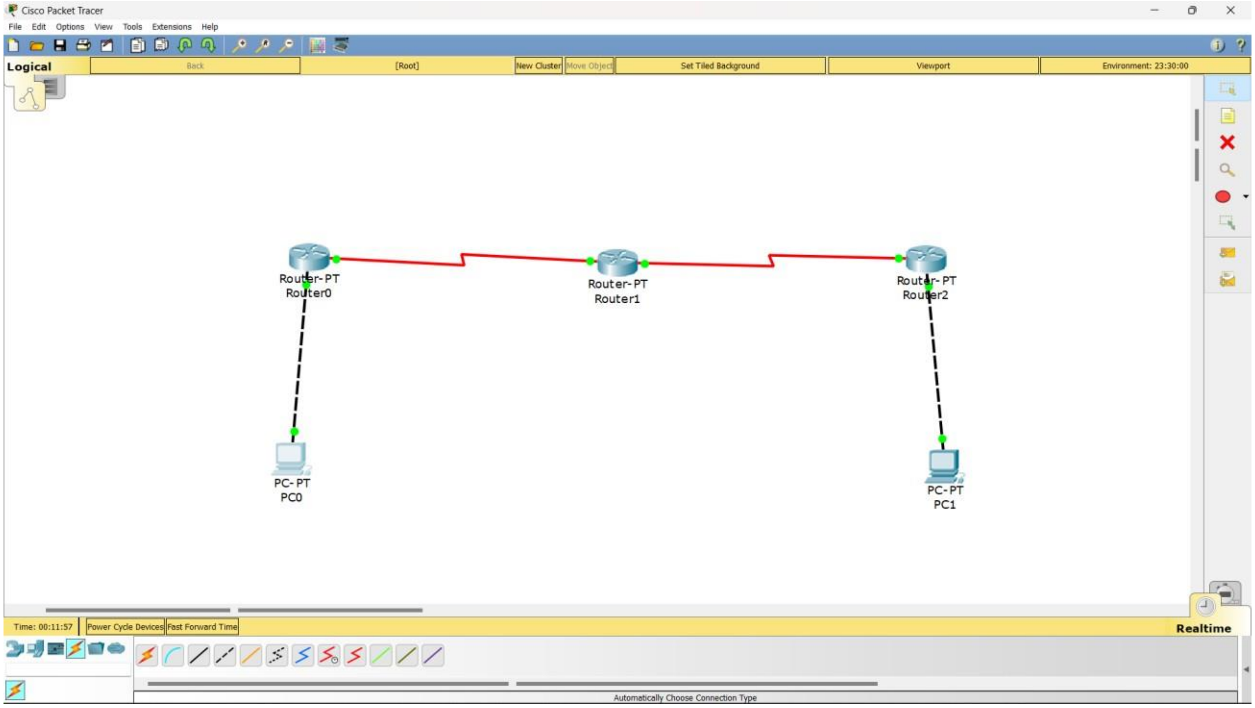
```
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.6

Pinging 10.0.0.6 with 32 bytes of data:

Reply from 10.0.0.6: bytes=32 time=1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128

Ping statistics for 10.0.0.6:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>
```

Experiment No 5

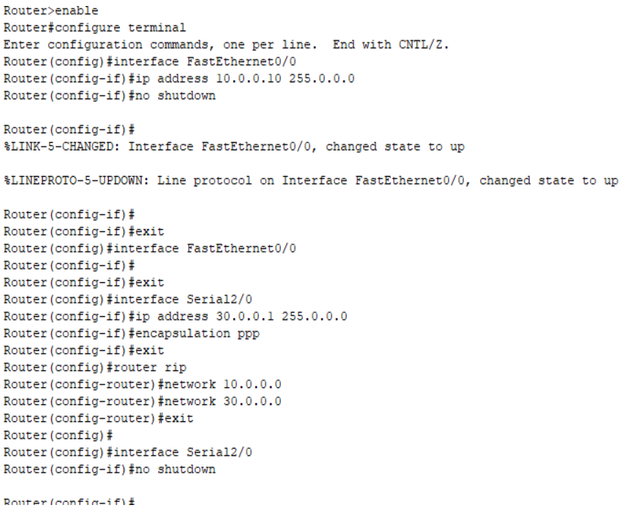
Aim of the program

Configuring RIP Routing Protocol in Routers

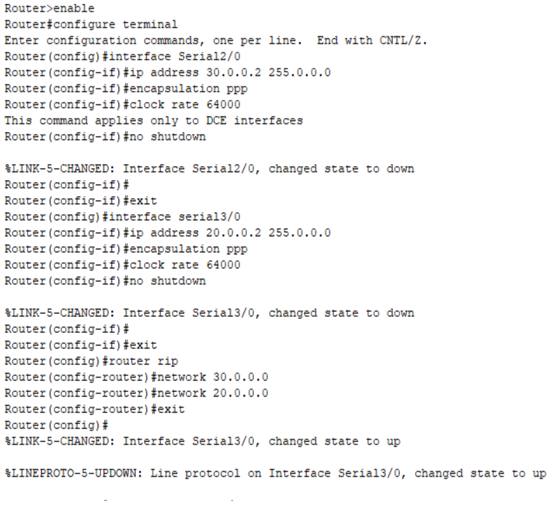
Topology



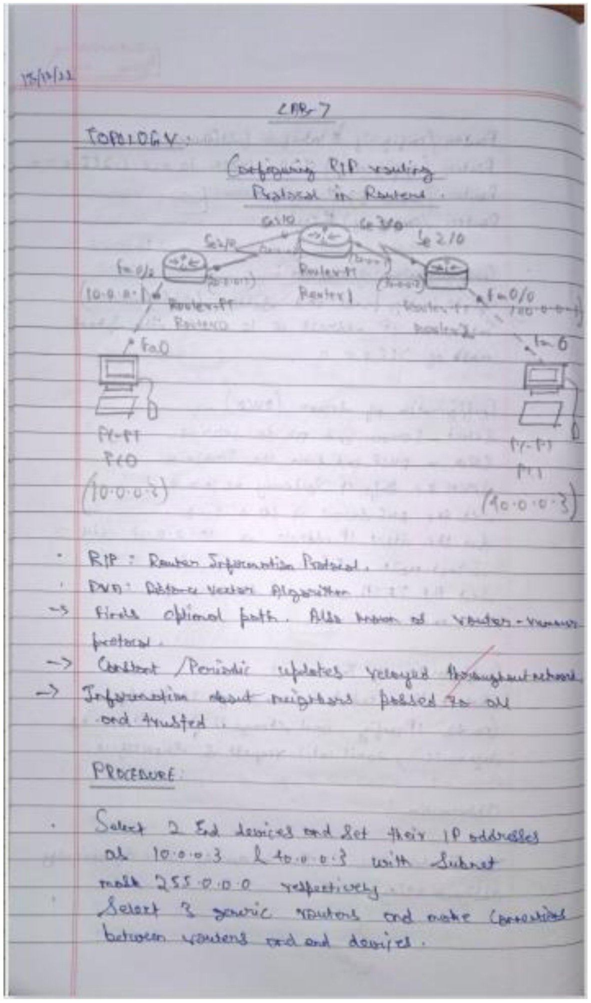
Procedure



```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface FastEthernet0/0
Router(config-if)#ip address 10.0.0.10 255.0.0.0
Router(config-if)#no shutdown

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

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

Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface Serial2/0
Router(config-if)#ip address 30.0.0.1 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 10.0.0.0
Router(config-router)#network 30.0.0.0
Router(config-router)#exit
Router(config)#
Router(config)#interface Serial2/0
Router(config-if)#no shutdown

Router(config-if)#
```



```
Router>enable
Router#configure terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface Serial2/0
Router(config-if)#ip address 30.0.0.2 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#clock rate 64000
This command applies only to DCE interfaces
Router(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial2/0, changed state to down
Router(config-if)#
Router(config-if)#exit
Router(config)#interface serial3/0
Router(config-if)#ip address 20.0.0.2 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown

%LINK-5-CHANGED: Interface Serial3/0, changed state to down
Router(config-if)#
Router(config-if)#exit
Router(config)#router rip
Router(config-router)#network 30.0.0.0
Router(config-router)#network 20.0.0.0
Router(config-router)#exit
Router(config)#
%LINK-5-CHANGED: Interface Serial3/0, changed state to up

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



13/12/22

LAB-7

TOPOLOGY :

Configuring RIP routing Protocol in Routers.

Se2/0
Se3/0
Se3/0
Se2/0
fa0/0
(10.0.0.1)
(20.0.0.1)
Router-PT
Router0
Router-PT
Router1
(30.0.0.1)
(30.0.0.2)
Router-PT
Router2
fa0/0
(40.0.0.1)
Fa0
Fa0
PC-PT
PC0
(10.0.0.3)
PC-PT
PC1
(40.0.0.3)

- RIP : Router Information Protocol.
- DVA : Distance Vector Algorithm

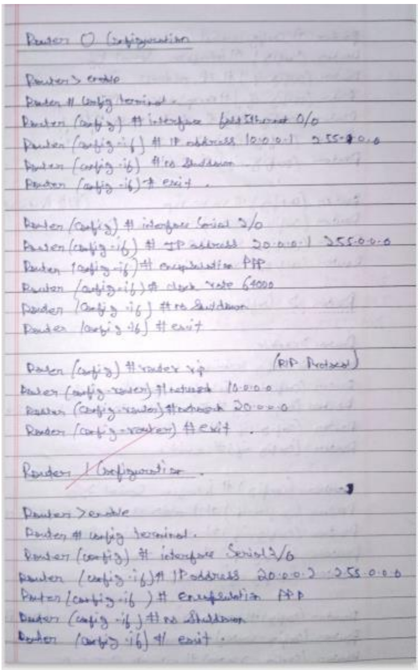
-> Finds optimal path. Also known as router-rumour protocol.

-> Constant / Periodic updates relayed throughout network.

-> Information about neighbors passed to all and trusted.

PROCEDURE :

- Select 2 End devices and set their IP addresses as 10.0.0.3 & 40.0.0.3 with Subnet mask 255.0.0.0 respectively.
- Select 3 generic routers and make connections between routers and end devices.

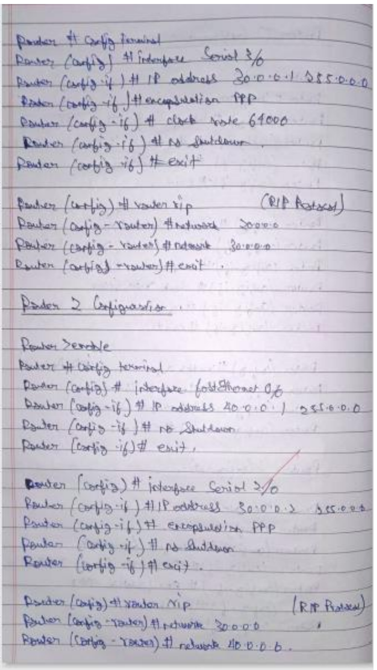


Router 0 Configuration

```
Router> enable
Router # Config terminal.
Router (config) # interface fastEthernet 0/0
Router (config-if) # IP address 10.0.0.1 255.0.0.0
Router (config-if) # no Shutdown
Router (config-if) # exit .

Router (config) # interface Serial 2/0
Router (config-if) # IP address 20.0.0.1 255.0.0.0
Router (config-if) # encapsulation PPP
Router (config-if) # clock rate 64000
Router (config-if) # no Shutdown
Router (config-if) # exit

Router (config) # router rip                (RIP Protocol)
Router (config-router) # network 10.0.0.0
Router (config-router) # network 20.0.0.0
Router (config-router) # exit .
```

Router 1 Configuration .

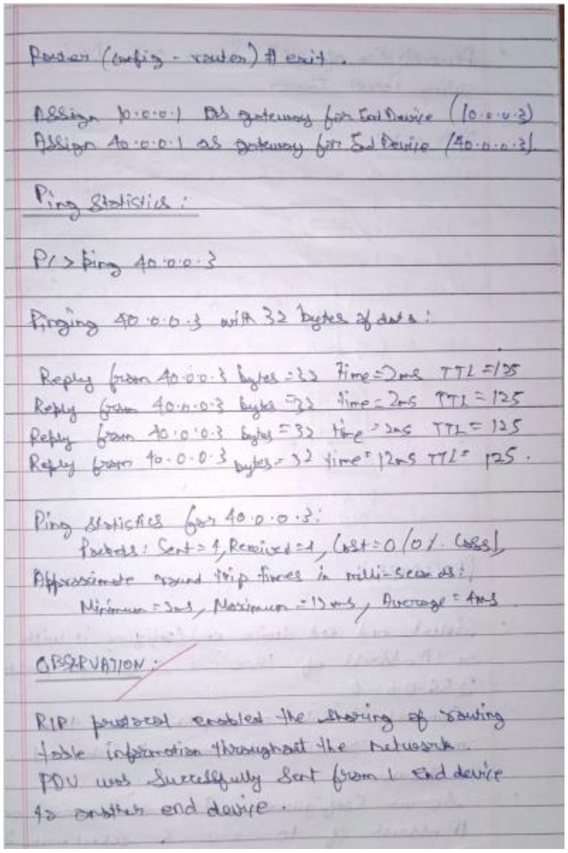
```
Router > enable
Router # config terminal.
Router (config) # interface Serial 2/0
Router (config-if) # IP address 20.0.0.2 255.0.0.0
Router (config-if) # encapsulation PPP
Router (config-if) # no Shutdown
Router (config-if) # exit .
```



```
Router # Config terminal
Router (config) # interface Serial 3/0
Router (config-if) # IP address 30.0.0.1 255.0.0.0
Router (config-if) # encapsulation PPP
Router (config-if) # clock rate 64000
Router (config-if) # no Shutdown .
Router (config-if) # exit

Router (config) # router rip               (RIP Protocol)
Router (config-router) # network 20.0.0.0
Router (config-router) # network 30.0.0.0
Router (config-router) # exit .
```

Router 2 Configuration .

```
Router > enable
Router # config terminal
Router (config) # interface fastEthernet 0/0
Router (config-if) # IP address 40.0.0.1 255.0.0.0
Router (config-if) # no Shutdown
Router (config-if) # exit .

Router (config) # interface Serial 2/0
Router (config-if) # IP address 30.0.0.2 255.0.0.0
Router (config-if) # encapsulation PPP
Router (config-if) # no Shutdown
Router (config-if) # exit .

Router (config) # router rip               (RIP Protocol)
Router (config-router) # network 30.0.0.0
Router (config-router) # network 40.0.0.0 .
```



Router (config - router) # exit.

Assign 10.0.0.1 as gateway for End Device (10.0.0.3)
Assign 40.0.0.1 as gateway for End Device (40.0.0.3).

Ping Statistics :

PC > Ping 40.0.0.3

Pinging 40.0.0.3 with 32 bytes of data :

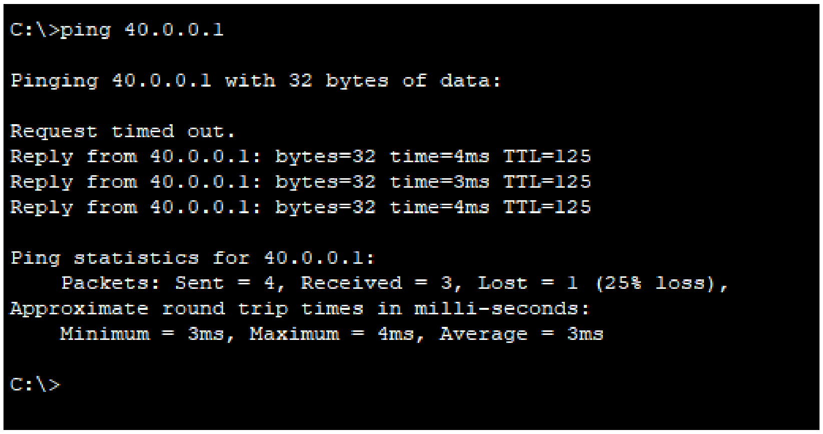
Reply from 40.0.0.3 bytes = 32 Time = 2ms TTL = 125
Reply from 40.0.0.3 bytes = 32 time = 2ms TTL = 125
Reply from 40.0.0.3 bytes = 32 time = 3ms TTL = 125
Reply from 40.0.0.3 bytes = 32 time = 12ms TTL = 125.

Ping statistics for 40.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0 (0% Loss),
Approximate round trip times in milli-seconds :
Minimum = 2ms, Maximum = 12ms, Average = 4ms

OBSERVATION :

RIP protocol enabled the sharing of routing table information throughout the Network.
PDU was successfully sent from 1 End device to another end device.

Output:



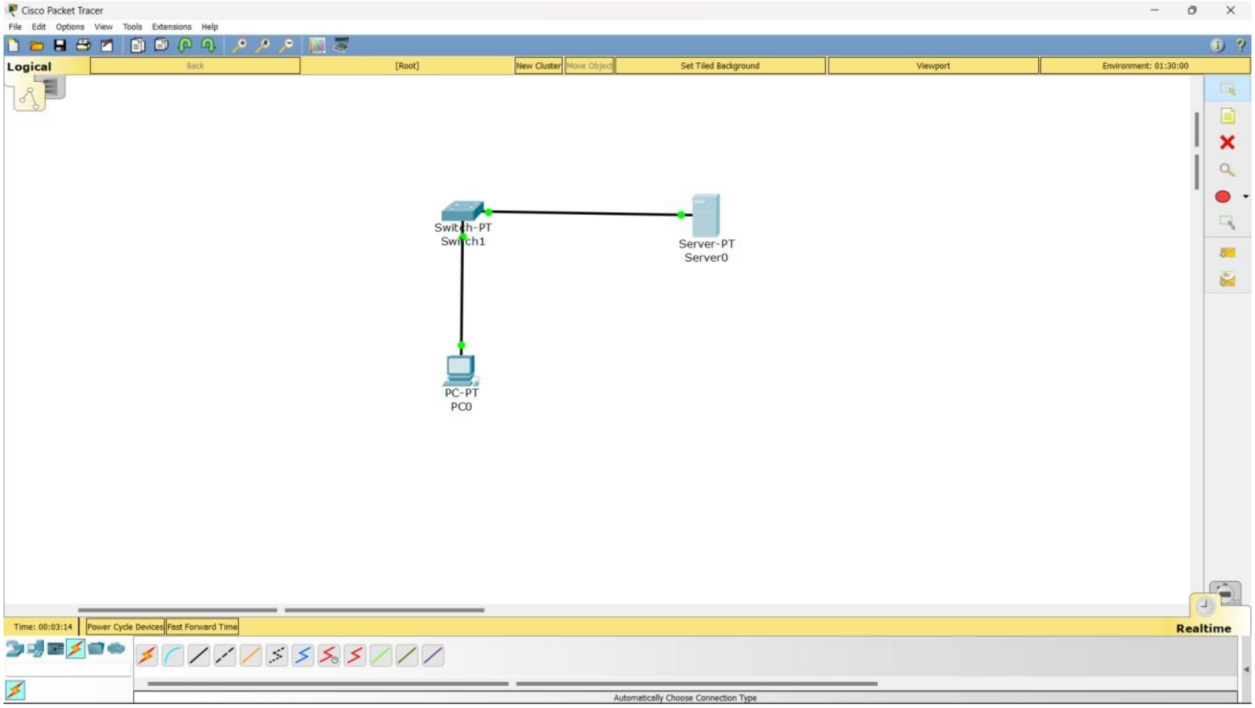
```
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 40.0.0.1: bytes=32 time=4ms TTL=125
Reply from 40.0.0.1: bytes=32 time=3ms TTL=125
Reply from 40.0.0.1: bytes=32 time=4ms TTL=125

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 = 4ms, Average = 3ms

C:\>
```

Experiment No 6

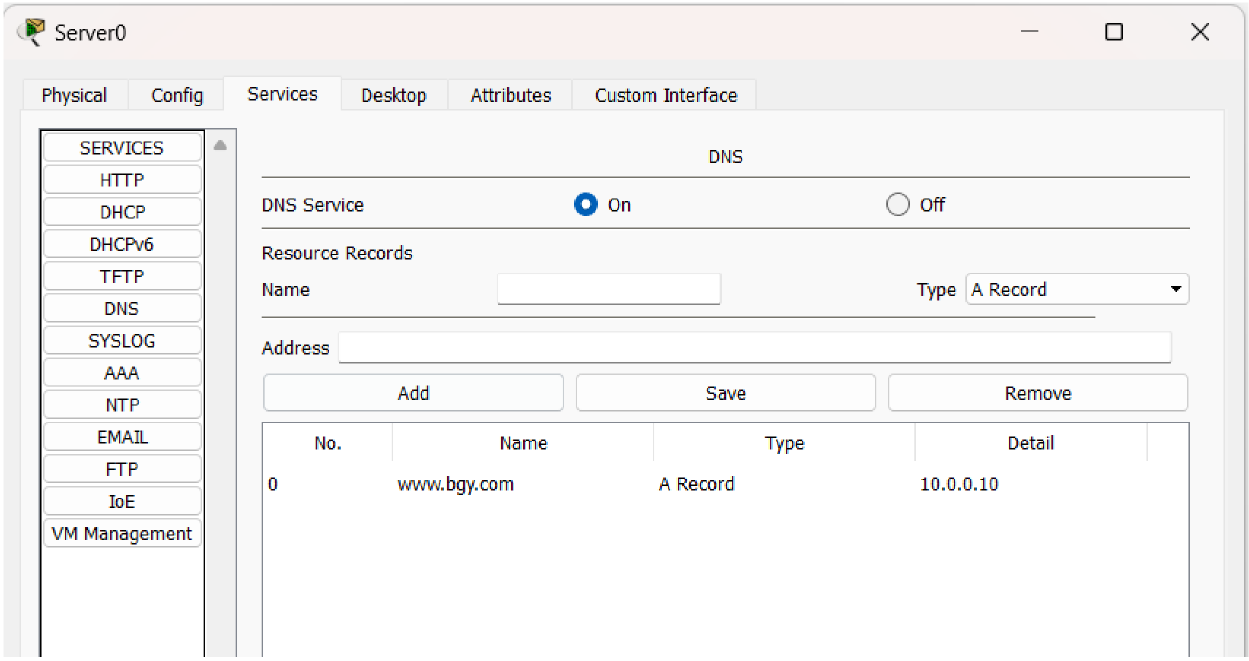
Aim of the program

Demonstration of WEB server and DNS using Packet Tracer

Topology



Procedure



DNS

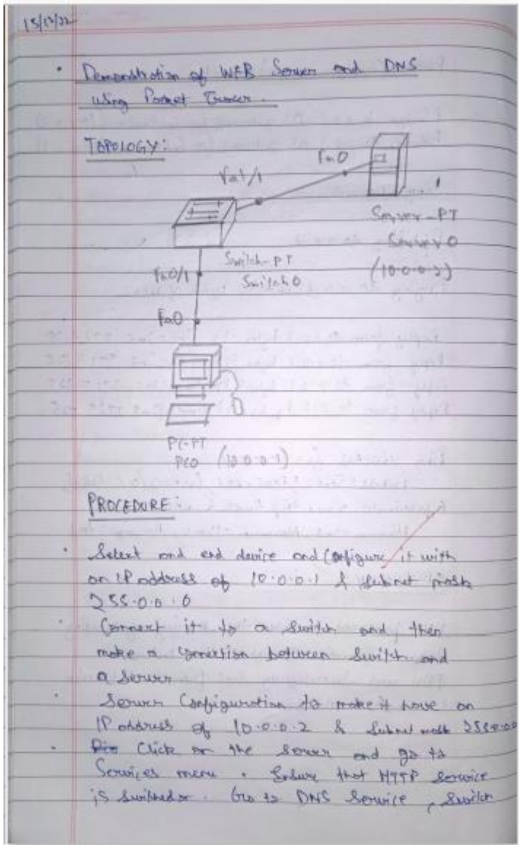
DNS Service ● On ○ Off

Resource Records

Name Type A Record

Address

No.	Name	Type	Detail
0	www.bgy.com	A Record	10.0.0.10



15/12/22

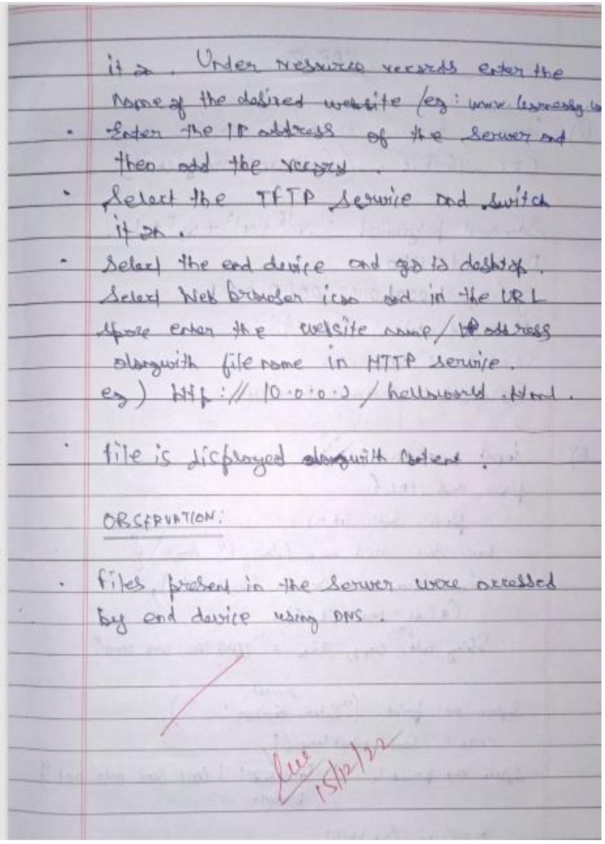
- Demonstration of WEB Server and DNS using Packet Tracer.

TOPOLOGY:

Fa0
Fa1/1
Server-PT
Server0
(10.0.0.2)
Switch-PT
Switch0
Fa0/1
Fa0
PC-PT
PC0 (10.0.0.1)

PROCEDURE:

- Select and end device and configure it with an IP address of 10·0·0·1 & subnet mask 255·0·0·0
- Connect it to a switch and then make a connection between switch and a server.
- Server Configuration to make it have an IP address of 10·0·0·2 & subnet mask 255·0·0·0
- ~~[illegible]~~ Click on the server and go to Services menu. Ensure that HTTP service is switched on. Go to DNS service, switch



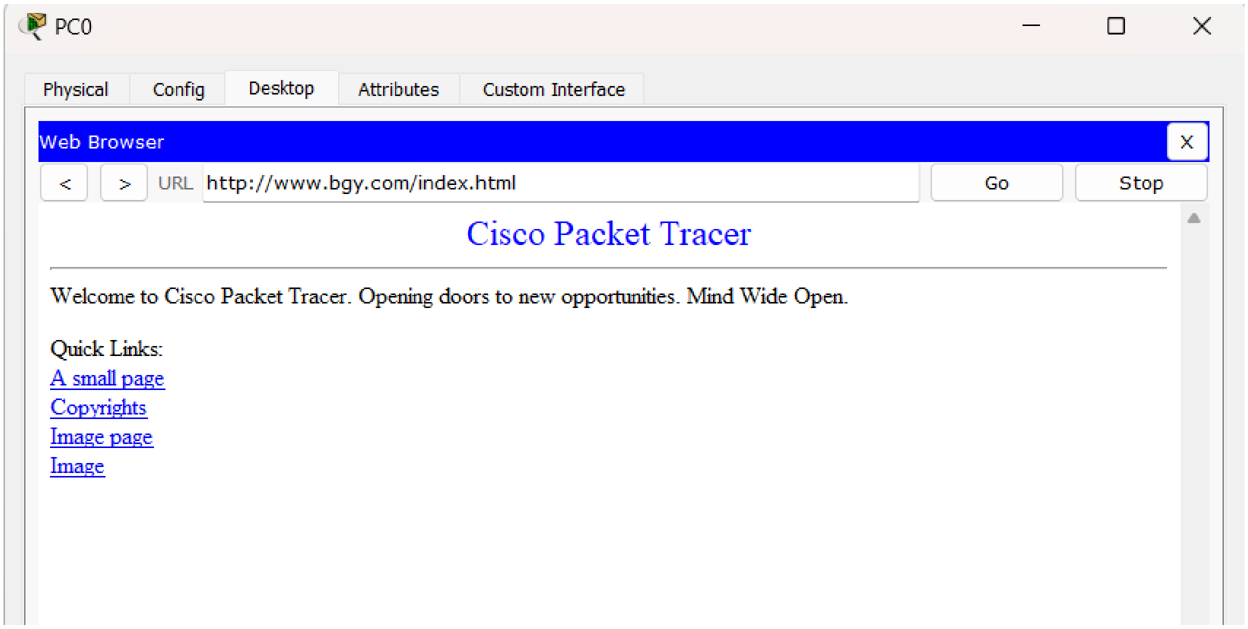
it on. Under resource records enter the name of the desired website (eg: www.[illegible]
- Enter the IP address of the server and then add the record.
- Select the TFTP service and switch it on.
- Select the end device and go to desktop. Select Web browser icon and in the URL space enter the website name/~~IP~~ address alongwith file name in HTTP service. eg) http://10·0·0·2/helloworld.html.
- file is displayed ~~alongwith~~ content.

OBSERVATION:

- Files present in the server were accessed by end device using DNS.

15/12/22

Output



http://www.bgy.com/index.html

Cisco Packet Tracer

Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.

Quick Links:
A small page
Copyrights
Image page
Image

Cycle-2

Experiment No 1

Aim of the Experiment

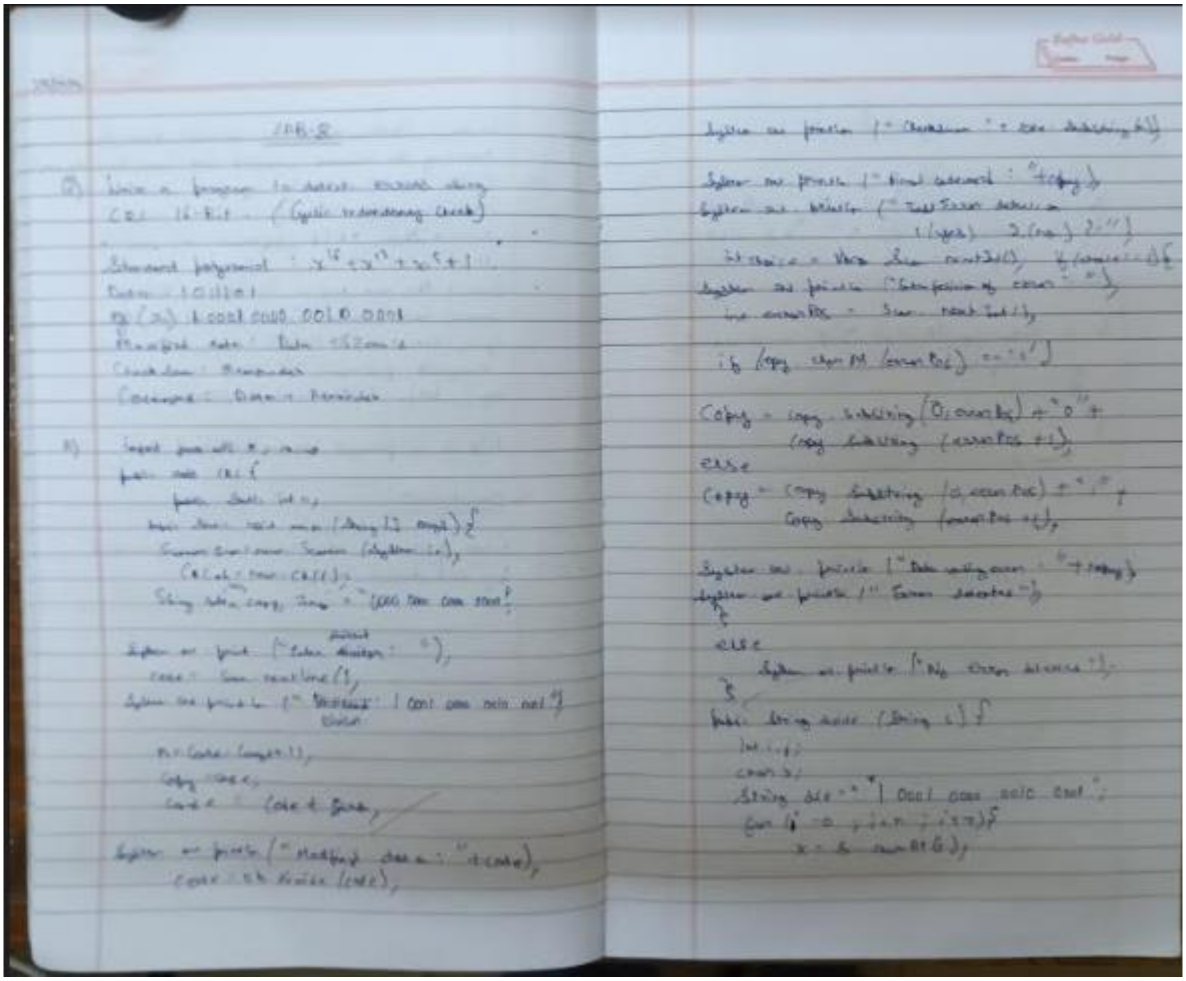
Write a program for error detecting code using CRC-CCITT (16-bits).

Code

```
#include
<iostream>
             #include <string>


             using namespace std;


             String divide(String s)
             {
                 int i,j;
                 char x;
                 String div = "10001000000100001";
                 for(i=0;i<7;i++)
                 {
                     x = s[i];
                     for(j=0;j<17;j++)
                     {
                         if(x=='1'){
                             if(s[i+j]!=div[j]){
                                 s = s.substr(0,i+j) + "1" + s.substr(i+j+1);
                             }
                             else{
                                 s = s.substr(0,i+j) + "0" + s.substr(i+j+1);
                             }
                         }
                     }
                 }
                 return s;
             }
             int main()
             {
                 int n;
                 int choice;
                 int errorPos;
```

```
    long data = 0;
    std::string divisor = "10001000000100001";
    std::string zero = "0000000000000000";
    std::string code,copy;
    cout<<"CRC 16-bit";
    cout<<"\nEnter the data(dividend)";
    cin>>code;
    cout<<"\nStandard polynomial(divisor/g(x)) is 10001000000100001";
    n = code.length();
    copy = code;
    code = code+zero;
    cout<<"\nModified data is"<<code;
    code = divide(code);
    cout<<"\nChecksum is "<<code.substr(n);
    cout<<"\nFinal codeword is "<<copy;
    cout<<"\nError checking 1(yes) 2(no)";
    cin>>choice;
    if(choice==1){
    cout<<"\nEnter error position: ";
    cin>>errorPos;
    if(copy[errorPos]=='1')
    {
        copy = copy.substr(0,errorPos) + "0" + copy.substr(errorPos+1);
    }
    else
    {
        copy = copy.substr(0,errorPos) + "1" + copy.substr(errorPos+1);
    }
    cout<<"\nData causing error"<<copy;
    cout<<"\nError detected";
    }
    else{
        cout<<"\nNo error found";
    }
    return 0;
}
```

Observation:

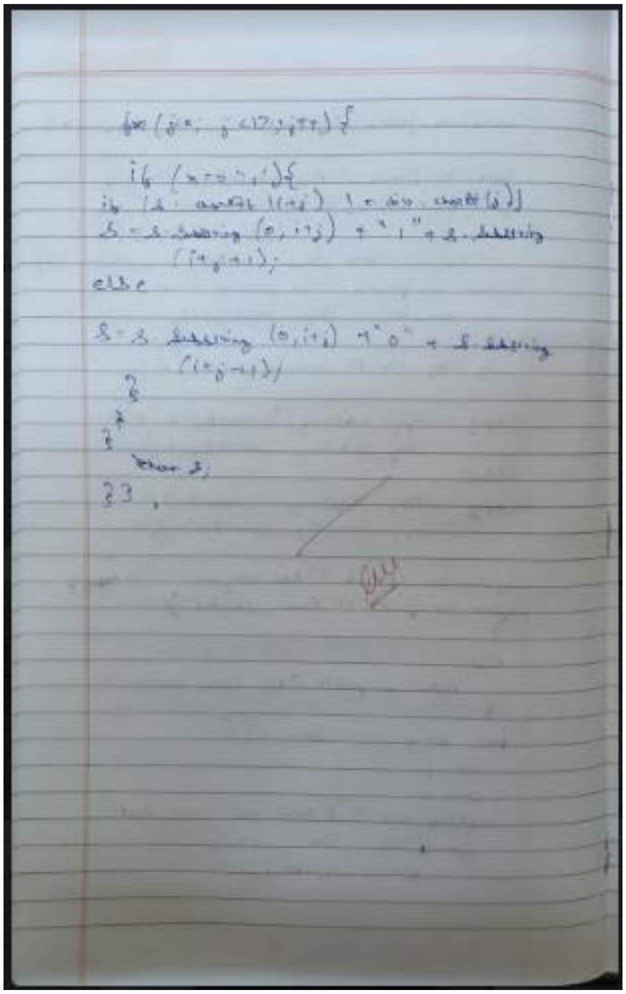


LAB-8

Q) Write a program to detect errors using CRC 16-Bit. (Cyclic redundancy check)

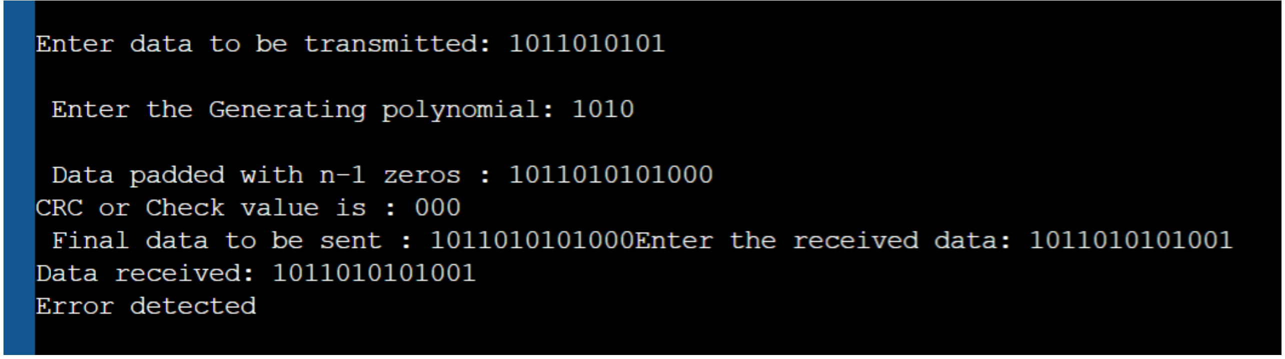
Standard polynomial : $x^{16} + x^{12} + x^5 + 1$
Data : 1011101
g(x) : 1 0001 0000 0010 0001
Transmitted data : Data + 16 zeros
Check sum : Remainder
Codeword : Data + Remainder

```
A)  import java.util.*;
    public class CRC {
        public static int n;
      public static void main (String[] args) {
       Scanner scan = new Scanner (System.in);
        CRC ob = new CRC();
       String data, copy, zeros = "0000 0000 0000 0000";

    System.out.print ("Enter division: ");
      code = scan.nextLine();
    System.out.println (" Generator: 1 0001 0000 0010 0001");

      n = code.length();
      copy = code;
      code += zeros;

    System.out.println ("Modified data: " + code);
       code = ob.divide (code);

System.out.println ("Checksum: " + code.substring(n));

System.out.println ("Final codeword: " + copy);
System.out.println (" Test Error detection
                         1(yes) 2(no) ?: ");
  int choice = new Scan.nextInt(); if (choice == 1) {
System.out.println ("Enter position of error: ");
  int errorPos = Scan.nextInt();

 if (copy.charAt (errorPos) == '0')

Copy = copy.substring (0, errorPos) + "0" +
         copy.substring (errorPos + 1);
else
Copy = copy.substring (0, errorPos) + "1" +
         copy.substring (errorPos + 1);

System.out.println (" Data with error: " + copy);
System.out.println (" Error detected ");
}
 else
    System.out.println ("No error detected");
 }
public String divide (String s) {
  int i, j;
  char x;
  String div = "1 0001 0000 0010 0001";
  for (i = 0; i < n; i++) {
      x = s.charAt(i);
```



```
for (j=0; j<17;j++) {
  if (x=='1'){
if (s.charAt(i+j) != div.charAt(j))
s = s.substring (0, i+j) + "1" + s.substring
      (i+j+1);
else
s = s.substring (0,i+j) +"0" + s.substring
      (i+j+1);
  }
 }
}
  return s;
}}.
```

Output



```
Enter data to be transmitted: 1011010101

 Enter the Generating polynomial: 1010

 Data padded with n-1 zeros : 1011010101000
CRC or Check value is : 000
 Final data to be sent : 1011010101000Enter the received data: 1011010101001
Data received: 1011010101001
Error detected
```

Experiment No 2

Aim of the Experiment

Write a program for distance vector algorithm to find suitable path for transmission.

Code

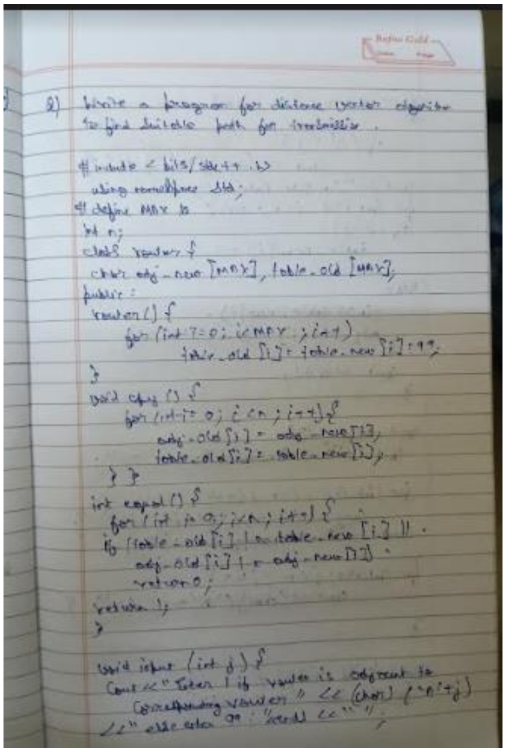
```
#include
<bits/stdc++.h>

                  using namespace std;

                  #define MAX 10

                  int n;

                  class router {

                  char adj_new[MAX], adj_old[MAX];

                  int table_new[MAX], table_old[MAX];

                   public:

                   router( ){

                  for(int i=0;i<MAX;i++) table_old[i]=table_new[i]=99;

                   }

                  void copy( ){

                  for(int i=0;i<n;i++) {

                   adj_old[i] =adj_new[i];

                   table_old[i]=table_new[i];

                   }

                   }

                  int equal( ) {

                  for(int i=0;i<n;i++)
```

```
if(table_old[i]!=table_new[i]||adj_new[i]!=adj_old[i])return 0;
return 1;
 }
void input(int j) {
 cout<<"Enter 1 if the corresponding router is adjacent to router"
<<(char)('A'+j)<<" else enter 99: "<<endl<<" ";
for(int i=0;i<n;i++)
if(i!=j) cout<<(char)('A'+i)<<" ";
 cout<<"\nEnter matrix:";
for(int i=0;i<n;i++) {
if(i==j)
 table_new[i]=0;
else
 cin>>table_new[i];
 adj_new[i]= (char)('A'+i);
 }
 cout<<endl;
 }
void display(){
 cout<<"\nDestination Router: ";
for(int i=0;i<n;i++) cout<<(char)('A'+i)<<" ";
 cout<<"\nOutgoing Line: ";
```

```
for(int i=0;i<n;i++) cout<<adj_new[i]<<" ";

 cout<<"\nHop Count: ";

for(int i=0;i<n;i++) cout<<table_new[i]<<" ";

 }

void build(int j) {

for(int i=0;i<n;i++)

for(int k=0;(i!=j)&&(k<n);k++)

if(table_old[i]!=99)

if((table_new[i]+table_new[k])<table_new[k]) {

 table_new[k]=table_new[i]+table_new[k];

 adj_new[k]=(char)('A'+i);

 }

 }

} r[MAX];

void build_table( ) {

int i=0, j=0;

while(i!=n) {

for(i=j;i<n;i++) {

 r[i].copy();

 r[i].build(i);

 }

for(i=0;i<n;i++)
```

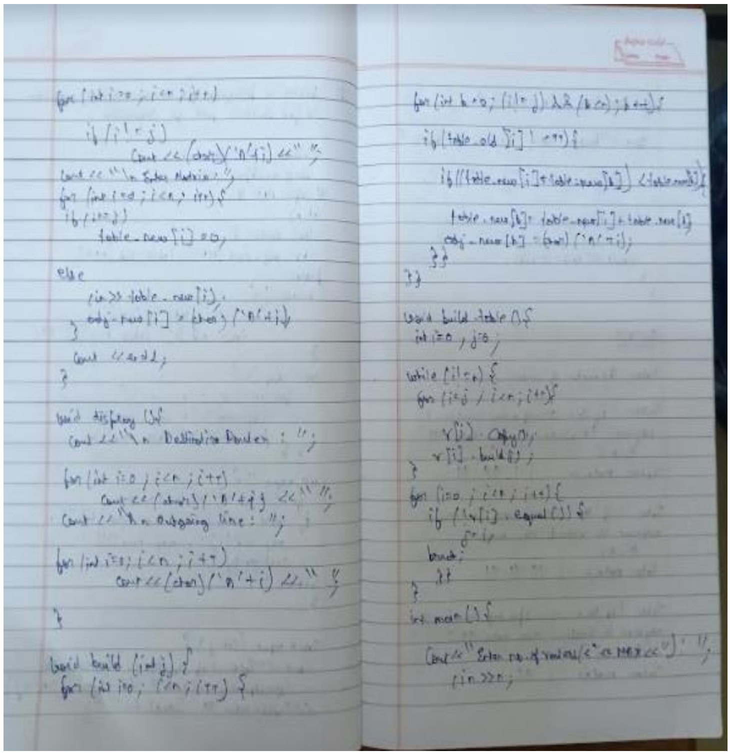
```
if(!r[i].equal()) {
 j=i;
break;
 }
 }
}
int main() {
 cout<<"Enter the number the routers(<"<<MAX<<"): "; cin>>n;
for(int i=0;i<n;i++) r[i].input(i);
 build_table();
for(int i=0;i<n;i++) {
 cout<<"Router Table entries for router "<<(char)('A'+i)<<":-";
 r[i].display();
 cout<<endl<<endl;
 }
}
```

Observation:



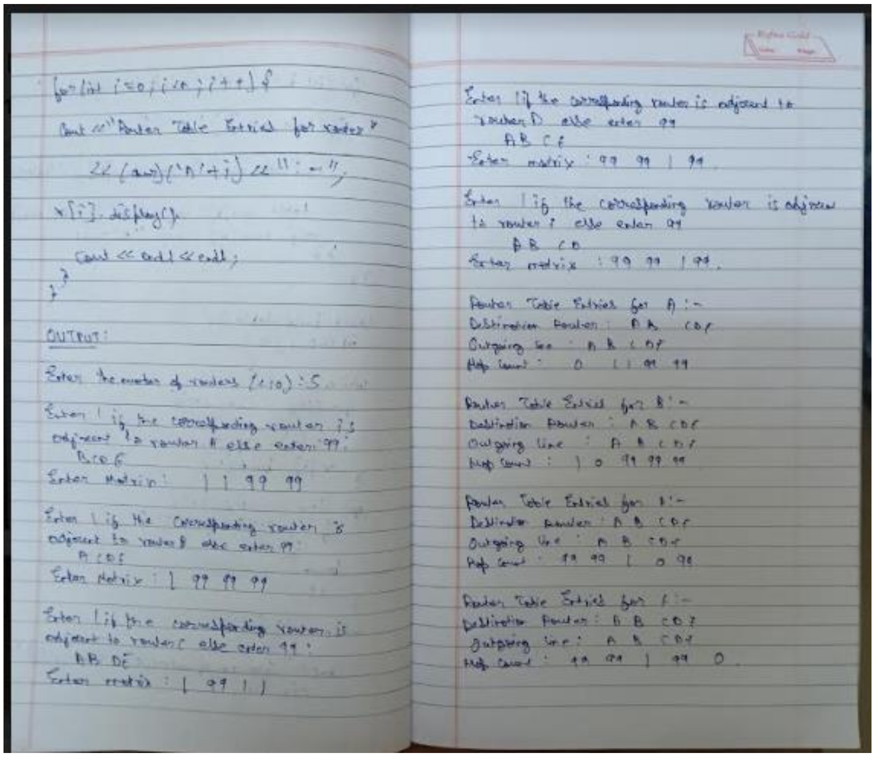
8) Write a program for distance vector algorithm to find suitable path for transmission.

```
# include < bits/stdc++.h>
 using namespace std;
# define MAX 10
int n;
class router {
 char adj_new [MAX], table_old [MAX];
public :
 router() {
      for (int i=0; i<MAX ; i++)
             table_old [i]= table_new [i]=99;
 }
 void copy () {
      for (int i= 0; i<n ; i++){
          adj_old [i] = adj_new [i];
          table_old [i] = table_new [i];
   } }
 int equal () {
   for (int i= 0; i<n; i++) {
  if (table_old [i] != table_new [i] ||
      adj_old [i] != adj_new [i])
      return 0;
 return 1;
 }

 void input (int j) {
  Cout << " Enter 1 if router is adjacent to
      Corresponding router " << (char) ('A'+j)
 <<" else enter 99 : "<<endl << " ";
```



```
for (int i=0 ; i<n ; i++)

    if (i != j)
          Cout << (char)('A'+i) <<" ";
Cout << "\n Enter Matrix : ";
for (int i=0 ; i<n ; i++) {
 if (i==j)
      table_new [i] =0;

else
    cin>> table_new [i];
    adj_new [i] = (char) ('A'+i);
 }
  Cout <<endl;
}

void display (){
  Cout <<"\n Destination Router : ";

 for (int i=0 ; i<n ; i++)
        Cout << (char) ('A'+i) << " ";
 Cout << "\n Outgoing line : ";

for (int i=0; i<n ; i++)
       Cout << (char) ('A'+i) << " ";

}

void build (int j) {
 for (int i=0; i<n; i++) {
```

```
for (int k =0; (i!= j) && (k<n) ; k++){
  if (table_old [i] != 99) {
    if ((table_new [i]+table_new[k]) <table_new[k]){
      table_new [k]= table_new [i] + table_new [i]
     adj_new [k] = (char) ('A'+i);
   }}
}}

void build table (){
 int i=0 , j=0 ;

while (i!=n) {
 for (i=j ; i<n; i++){
      r[i] . copy();
    r [i] . build() ;
 }
for (i=0 ; i<n ; i++) {
  if (!r[i] . equal ()) {
      j=i;
  break;
    }}
}
int main () {

  Cout << " Enter no. of routers(<" << MAX << ") : ";
      cin>>n;
```



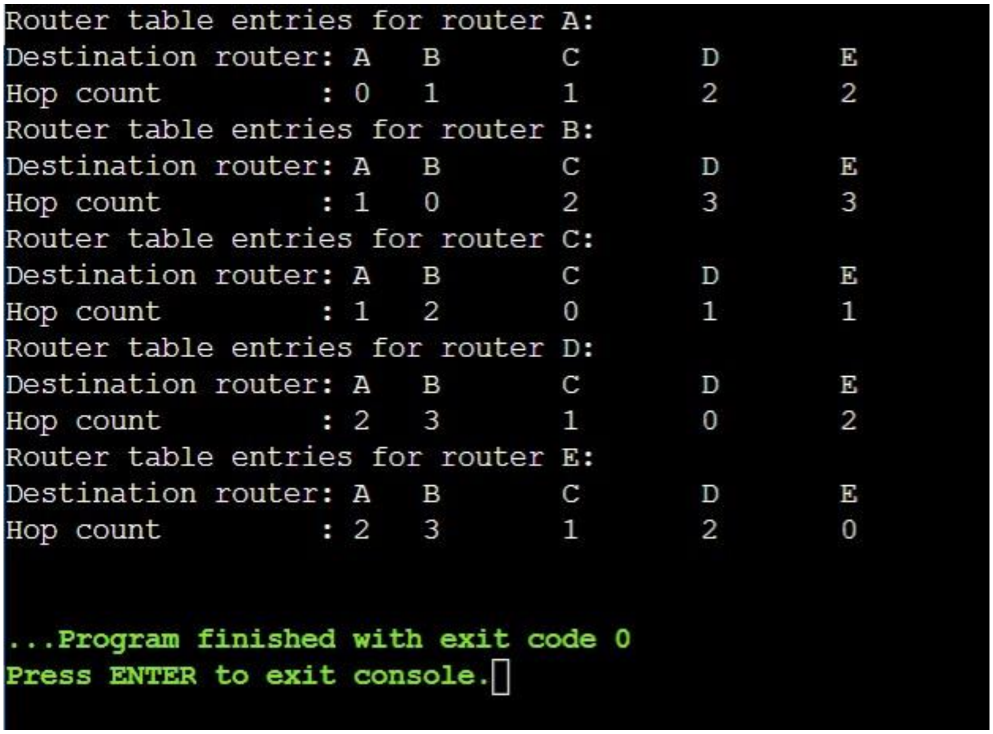
```
for (int i=0; i<n ; i++) {

 Cout <<"Router Table Entries for router "

     << (char)('A'+i) << " : - ";

r[i]. display();

  Cout << endl << endl;
}
}
```

<u>OUTPUT</u>:

```
Enter the number of routers (<10) : 5

Enter 1 if the corresponding router is
adjacent to router A else enter 99 :
   B C D E
Enter Matrix :     1 1 99 99

Enter 1 if the corresponding router is
adjacent to router B else enter 99
   A C D E
Enter Matrix : 1 99 99 99

Enter 1 if the corresponding router is
adjacent to router C else enter 99 :
   A B D E
Enter matrix : 1 99 1 1
```

```
Enter 1 if the corresponding router is adjacent to
router D else enter 99
   A B C E
Enter matrix : 99 99 1 99 .

Enter 1 if the corresponding router is adjacent
to router E else enter 99
   A B C D
Enter matrix : 99 99 1 99.

Router Table Entries for A :-
Destination Router :  A B  C D E
Outgoing line  :  A B C D E
Hop count :    0   1 1 99 99

Router Table Entries for B :-
Destination Router :  A B C D E
Outgoing line  :  A B C D E
Hop Count  :  1 0  99 99 99

Router Table Entries for D :-
Destination Router : A B C D E
Outgoing line : A B C D E
Hop Count :  99 99  1  0 99

Router Table Entries for E :-
Destination Router : A B C D E
Outgoing line :  A B C D E
Hop Count :  99 99  1  99  0 .
```

Output:



```
Router table entries for router A:
Destination router: A   B       C       D       E
Hop count         : 0   1       1       2       2
Router table entries for router B:
Destination router: A   B       C       D       E
Hop count         : 1   0       2       3       3
Router table entries for router C:
Destination router: A   B       C       D       E
Hop count         : 1   2       0       1       1
Router table entries for router D:
Destination router: A   B       C       D       E
Hop count         : 2   3       1       0       2
Router table entries for router E:
Destination router: A   B       C       D       E
Hop count         : 2   3       1       2       0


...Program finished with exit code 0
Press ENTER to exit console.
```

Experiment No 3

Aim of the Experiment

Implement Dijkstra’s algorithm to compute the shortest path for a given topology.

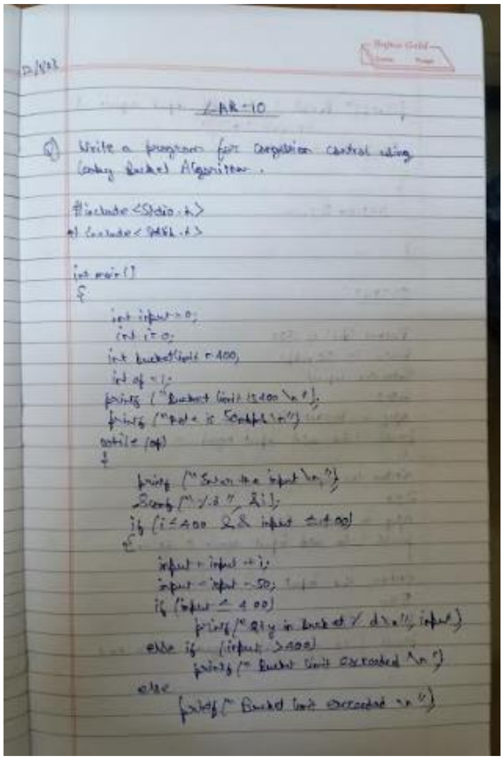
Code

```
#include
<iostream>


             using namespace std;


             void dijkstra(int arr[10][10],int N)
             {
                 int distance[N];
                 int predefine[N];
                 int visited[N];
                 int startnode,nextnode;
                 int count,min_dist;
                 int i,j;
                 cout<<"\nEnter the node to start with: ";
                 cin>>startnode;
                 for(i=0;i<N;i++)
                 {
                     distance[i] = arr[startnode][i];
                     predefine[i] = startnode;
                     visited[i] = 0;
                 }
                 distance[startnode] = 0;
                 visited[startnode] = 1;
                 count = 1;
                 while(count<N-1)
                 {
                     min_dist = 1000;
                     for(i=0;i<N;i++)
                     {
                         if(distance[i]<min_dist && visited[i]==0)
                         {
                             min_dist = distance[i];
                             nextnode = i;
                         }
                     }
                     visited[nextnode] = 1;
```

```
        for(i=0;i<N;i++)
        {
            if(visited[i] == 0)
            {
                if((min_dist + arr[nextnode][i]) < distance[i])
                {
                    distance[i] = min_dist + arr[nextnode][i];
                    predefine[i] = nextnode;
                }
            }
        }
        count = count + 1;
    }
    for(i=0;i<N;i++)
    {
        if(i!=startnode)
        {
            cout<<"\nDistance from node "<<i<<"="<<distance[i];
            cout<<"\nOptimal path is "<<i;
            j = i;
            do
            {
                j = predefine[j];
                cout<<" <- "<<j;
            }while(j!=startnode);
        }
    }
}
int main()
{
    int arr2[10][10];
    int N;
    cout<<"Dijkstra's algorithm";
    cout<<"\nEnter the number of nodes: ";
    cin>>N;
    cout<<"Enter weights/distances matrix for topology";
    for(int i = 0;i<N;i++)
    {
        for(int j = 0;j<N;j++)
        {
            cin>>arr2[i][j];
        }
    }
    dijkstra(arr2,N);
```

```
        return 0;
    }
```

Observation:

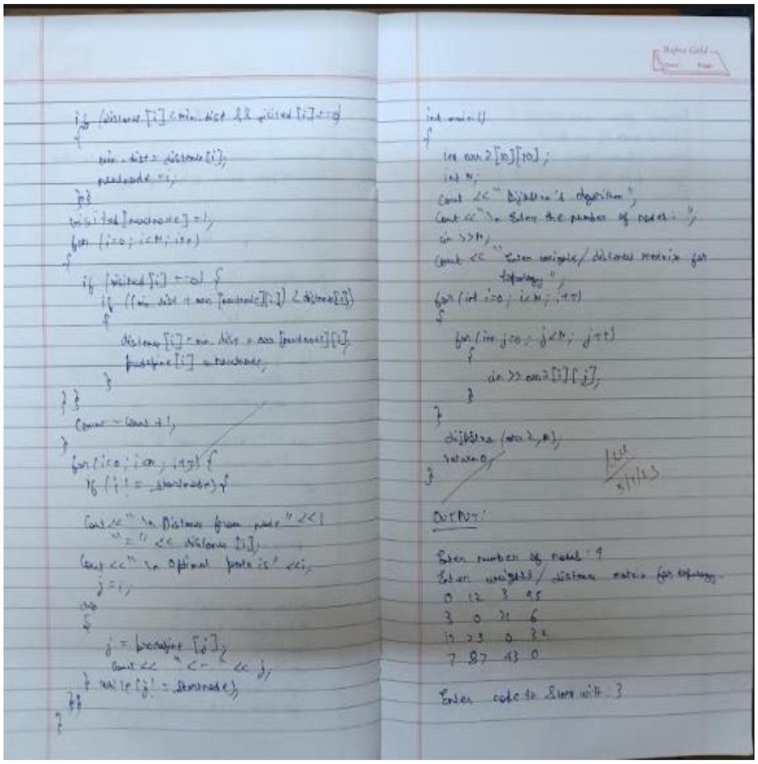


12/1/23

LAB-10

Q) Write a program for Congestion control using Leaky Bucket Algorithm.

```
#include<Stdio.h>
#include<Stdlib.h>

int main()
{
    int input=0;
    int i=0;
    int bucketlimit=400;
    int op=1;
    printf("Bucket limit is 400 \n");
    printf("Rate is 50pkt\n");
    while(op)
    {
        printf("Enter the input \n");
        Scanf("%d", &i);
        if(i<=400 && input <=400)
        {
            input = input +i;
            input = input -50;
            if(input <= 400)
                printf("Qty in bucket %d\n", input);
        else if (input >400)
                printf("Bucket limit exceeded \n")
        else
            printf("Bucket limit exceeded \n")
```



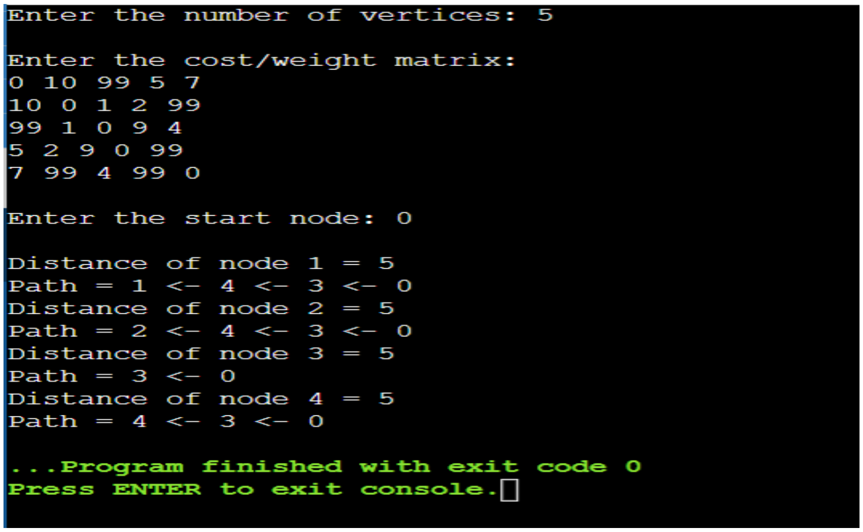
```
if (distance[i]<min_dist && visited[i]==0)
{
    min_dist = distance[i];
    nextnode =i;
}}
visited[nextnode]=1;
for (i=0;i<n;i++)
{
  if (visited[i]==0) {
    if ((min_dist + arr[nextnode][i]) < distance[i])
    {
        distance[i]=min_dist + arr[nextnode][i];
        pred[i] =nextnode;
    }
}}
count = count +1;
}
for(i=0;i<n;i++){
  if(i!=startnode){

  cout<<"\n Distance from node "<<i
      <<" = "<< distance [i];
  cout<<"\n Optimal path is" <<i;
    j=i;
  do
  {
     j = pred [j];
     cout<< "<-" << j;
  } while(j!=startnode);
}}
}
```

```
int main()
{
  int arr2[10][10];
  int n;
  cout <<"Dijkstra's Algorithm";
  cout<<"\n Enter the number of nodes: ";
  cin >>n;
  cout << "Enter weight / distance matrix for
            topology";
  for(int i=0; i<n; i++)
  {
    for(int j=0; j<n; j++)
     {
        cin >> arr2[i][j];
     }
  }
  dijkstra (arr2,n);
  return 0;
}
```

OUTPUT:

```
Enter number of nodes: 4
Enter weights / distance matrix for topology
 0  12  3  45
 3  0  22  6
 19 23  0  32
 7  87  43  0

Enter code to Start with: 3
```

Output:



```
Enter the number of vertices: 5

Enter the cost/weight matrix:
0 10 99 5 7
10 0 1 2 99
99 1 0 9 4
5 2 9 0 99
7 99 4 99 0

Enter the start node: 0

Distance of node 1 = 5
Path = 1 <- 4 <- 3 <- 0
Distance of node 2 = 5
Path = 2 <- 4 <- 3 <- 0
Distance of node 3 = 5
Path = 3 <- 0
Distance of node 4 = 5
Path = 4 <- 3 <- 0

...Program finished with exit code 0
Press ENTER to exit console.
```

Experiment No 4

Aim of the Experiment

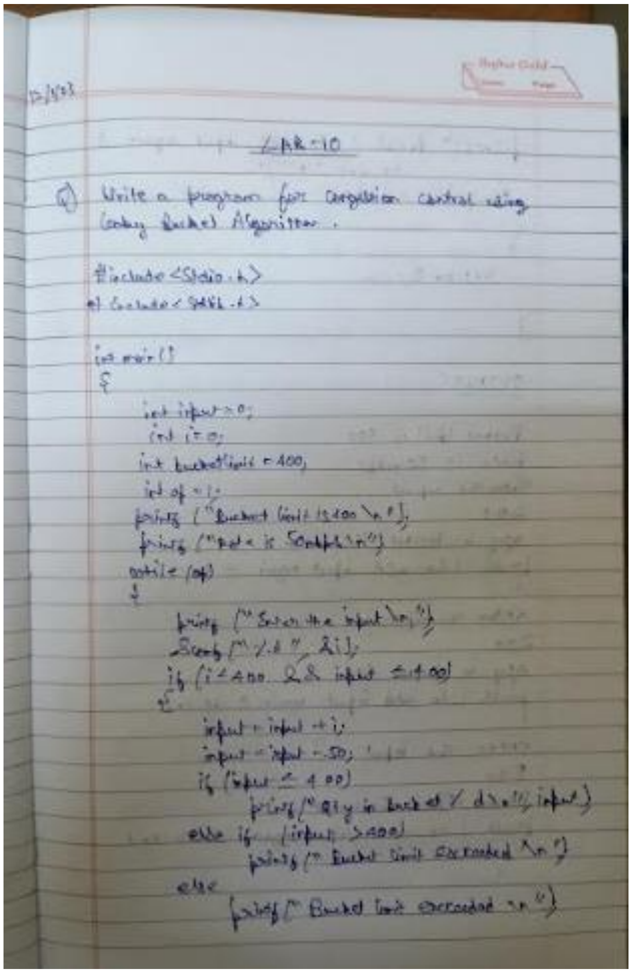
Write a program for congestion control using Leaky bucket algorithm

Code

```
#include<stdio.h>
                    #include<stdlib.h>
                    int main()
                    {
                        int input=0;
                        int i=0;
                        int bucketlimit=400;
                        int op=1;
                        printf("Bucket limit is 400\n");
                        printf("Rate is 50mbps\n");
                        while(op)
                        {
                            printf("enter the input\n");
                            scanf("%d",&i);
                            if(i<=400 && input<=400)
                            {

                                input=input+i;
                                input=input-50;
                                if(input<=400)
                                {
                                printf("qty in bucket %d\n",input);
                                }
                                else if(input>400)
                                {
                                 printf("Bucket limit Exceeded\n");
                                }
                            }
                            else
                            {
                                printf("Bucket limit Exceeded\n");
                            }



                            printf("press 1 to add input again 0 to end\n");
```

```
                scanf("%d",&op);
            }
            return 0;
        }
```

Observation:

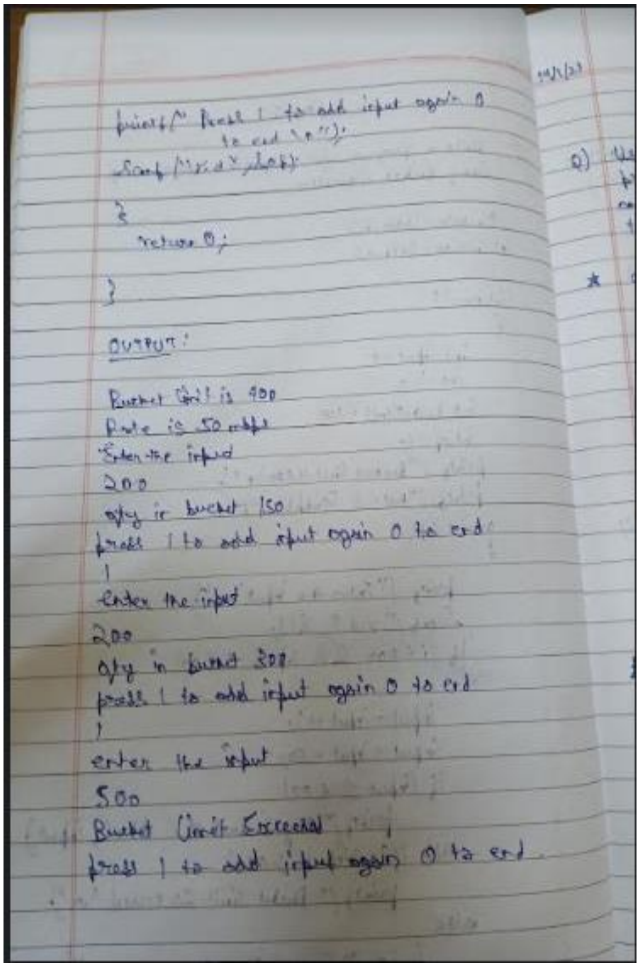


13/4/23

LAB-10

Q) Write a program for congestion control using Leaky Bucket Algorithm.

```
#include<stdio.h>
#include<stdlib.h>

int main()
{
    int input=0;
    int i=0;
    int bucketlimit=400;
    int op=1;
    printf("Bucket limit is 400 \n");
    printf("Rate is 50mbps\n");
    while(op)
    {
        printf("Enter the input\n");
        scanf("%d",&i);
        if(i<400 && input <=400)
        {
            input=input+i;
            input=input-50;
            if(input<=400)
                printf("Qty in bucket %d\n",input);
        else if (input>400)
                printf("Bucket limit Exceeded \n");
        else
            printf("Bucket limit exceeded \n");
```

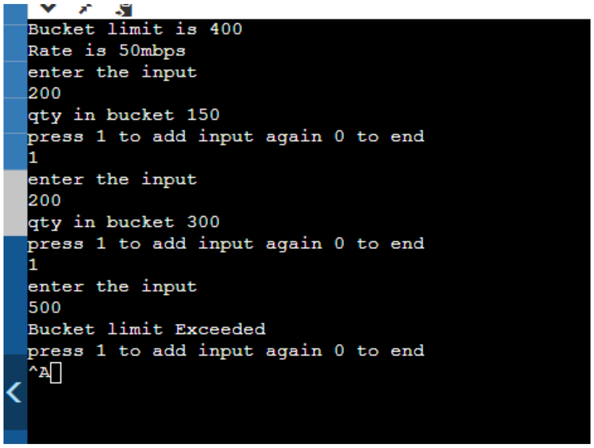


```
printf("Press 1 to add input again 0
        to end \n");
scanf("%d",&op);

}
  return 0;

}
```

OUTPUT:

```
Bucket limit is 400
Rate is 50 mbps
Enter the input
200
qty in bucket 150
press 1 to add input again 0 to end
1
Enter the input
200
Qty in bucket 300
press 1 to add input again 0 to end
1
enter the input
500
Bucket limit Exceeded
press 1 to add input again 0 to end
```

Output:



```
Bucket limit is 400
Rate is 50mbps
enter the input
200
qty in bucket 150
press 1 to add input again 0 to end
1
enter the input
200
qty in bucket 300
press 1 to add input again 0 to end
1
enter the input
500
Bucket limit Exceeded
press 1 to add input again 0 to end
^A
```

Experiment No 5

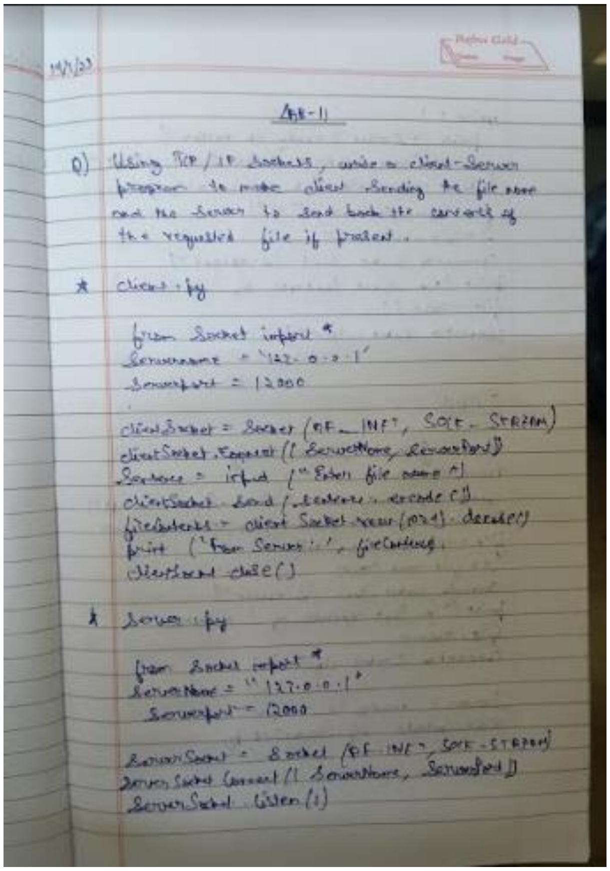
Aim of the Experiment

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.

Code

```
//Client:


            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()


            //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()
```

Observation:



19/1/23

LAB-11

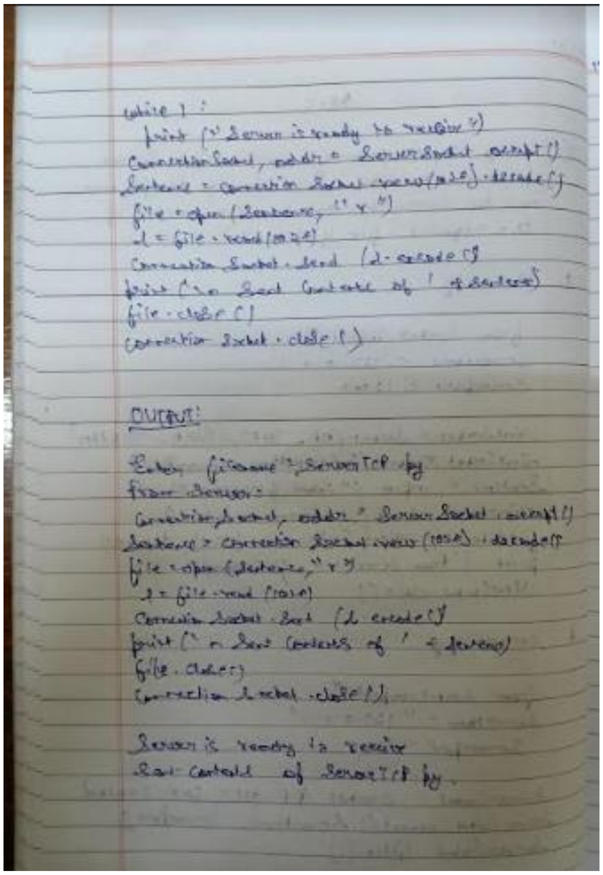
Q) Using TCP / IP Sockets, write a client-Server program to make client sending the file name and the Server to send back the contents of the requested file if present.

* client.py

```
from Socket import *
Servername = '127.0.0.1'
Serverport = 12000

clientSocket = Socket (AF_INET, SOCK_STREAM)
clientSocket.connect ((ServerName, ServerPort))
Sentence = input ("Enter file name:")
clientSocket.send (sentence.encode())
filecontents = client Socket.recv(1024).decode()
print ('From Server:', filecontents)
clientSocket.close()
```

* Server.py

```
from Socket import *
ServerName = "127.0.0.1"
 Serverport = 12000

ServerSocket = Socket (AF_INET, SOCK_STREAM)
ServerSocket Connect ((ServerName, ServerPort))
ServerSocket.listen(1)
```

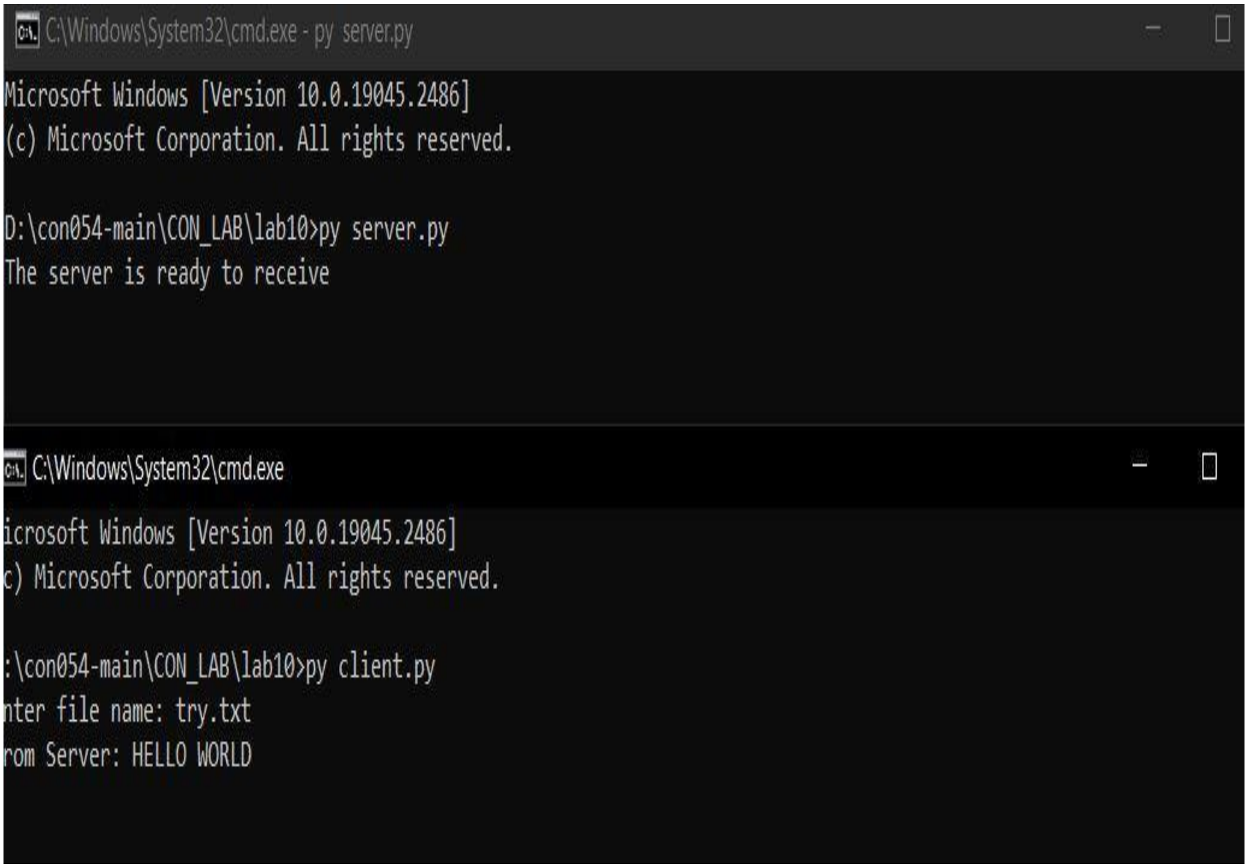


```
while 1:
  print ("Server is ready to receive")
ConnectionSocket, addr = ServerSocket.accept()
Sentence = Connection Socket.recv(1024).decode()
 file = open (Sentence, "r")
 l = file.read(1024)
 ConnectionSocket.send (l.encode())
print ('\n Sent Contents of ' + Sentence)
file.close()
Connection Socket.close()
```

OUTPUT:

```
Enter filename: ServerTCP.py
From Server:
 ConnectionSocket, addr = ServerSocket.accept()
Sentence = ConnectionSocket.recv(1024).decode()
file = open (Sentence, "r")
 l = file.read(1024)
 ConnectionSocket.Send (l.encode())
print ("\n Sent Contents of ' + Sentence)
 file.close()
 ConnectionSocket.close();

Server is ready to receive
Sent Contents of ServerTCP.py
```

Output



C:\Windows\System32\cmd.exe - py server.py

```
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

D:\con054-main\CON_LAB\lab10>py server.py
The server is ready to receive
```

C:\Windows\System32\cmd.exe

```
icrosoft Windows [Version 10.0.19045.2486]
c) Microsoft Corporation. All rights reserved.

:\con054-main\CON_LAB\lab10>py client.py
nter file name: try.txt
rom Server: HELLO WORLD
```

Experiment No 6

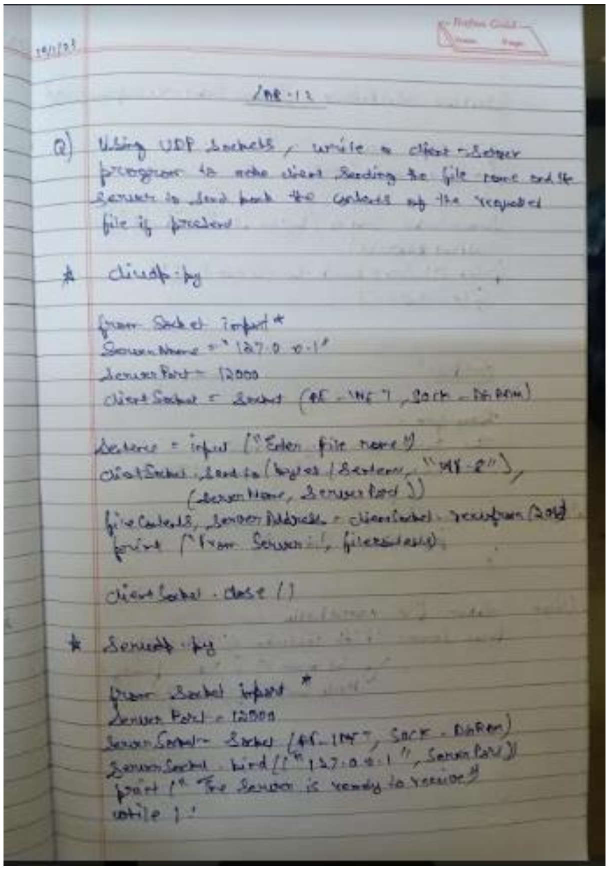
Aim of the Experiment

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.

Code

```
//Client:



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()



//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")
    l=file.read(2048)
    serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
    print ('\nSent contents of ', end = ' ')
    print (sentence)
# for i in sentence:
# print (str(i), end = '')
    file.close()
```

Observation:



19/1/23

LAB-12

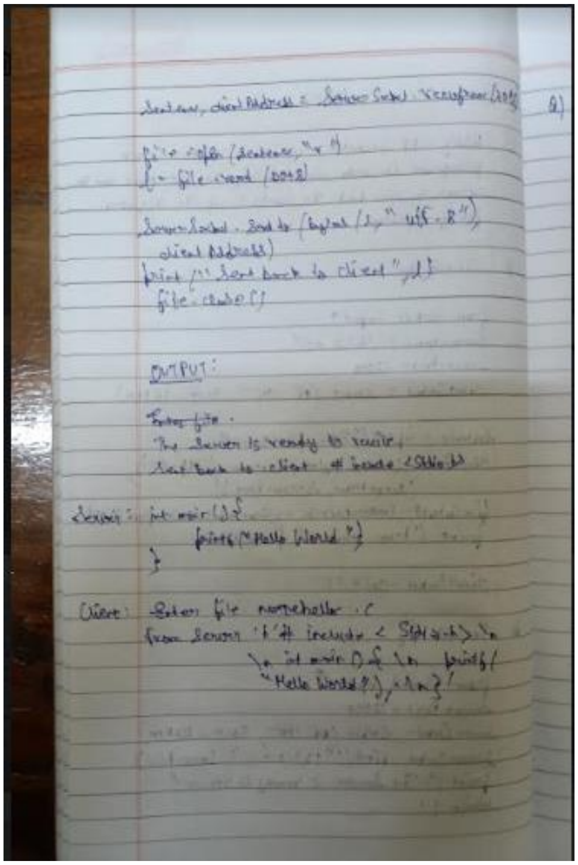
Q) Using UDP Sockets, write a client-Server program to make client Sending the file name and the Server to send back the contents of the requested file if present.

* clientb.py

```
from Socket import *
ServerName = "127.0.0.1"
ServerPort = 12000
clientSocket = Socket (AF_INET, SOCK_DGRAM)

Sentence = input ("Enter file name")
clientSocket.Sendto(bytes(Sentence, "utf-8"),
          (ServerName, ServerPort))
fileContents, ServerAddress = clientSocket.recvfrom(2048)
print ("From Server:", filecontents)

clientSocket.close()
```

* Serverb.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)

file = open(Sentence, "r")
l = file.read(2048)

ServerSocket.Sendto(bytes(l, "utf-8"),
  clientAddress)
print("Sent back to client", l)
 file.close()
```

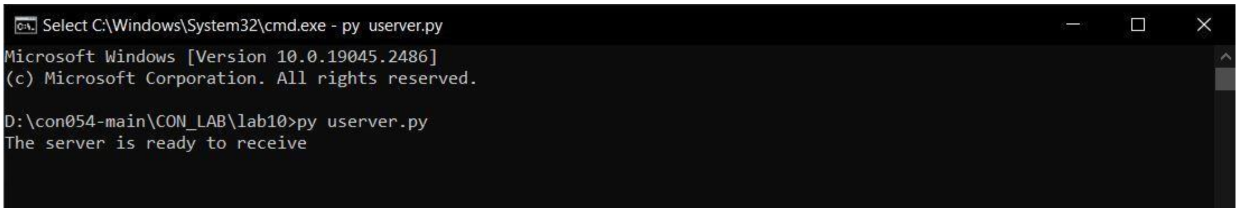
OUTPUT:

~~Enter file~~ .
The Server is ready to receive
Sent back to client #include <Stdio.h>

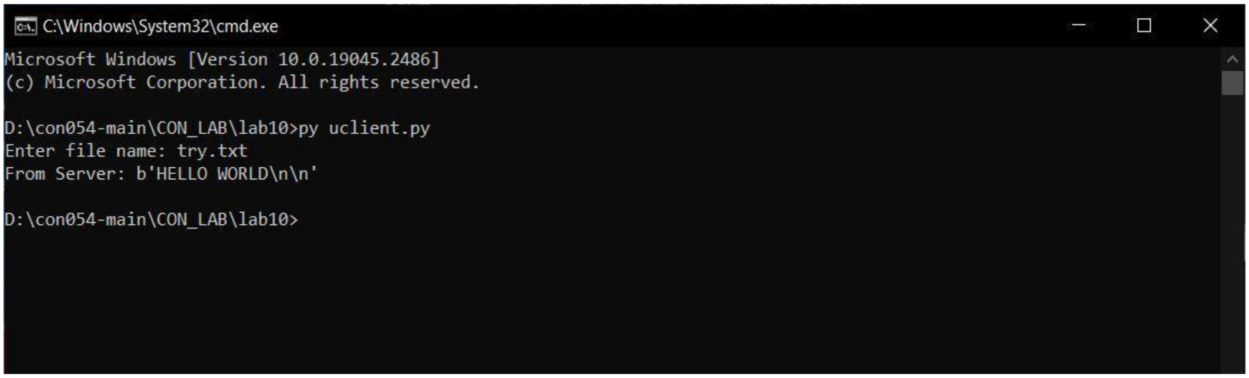
Server: int main() {
          printf("Hello World")
        }

Client: Enter file name hello.c
From Server b'#include <Stdio.h> \n \n int main() { \n printf("Hello World"); \n }'

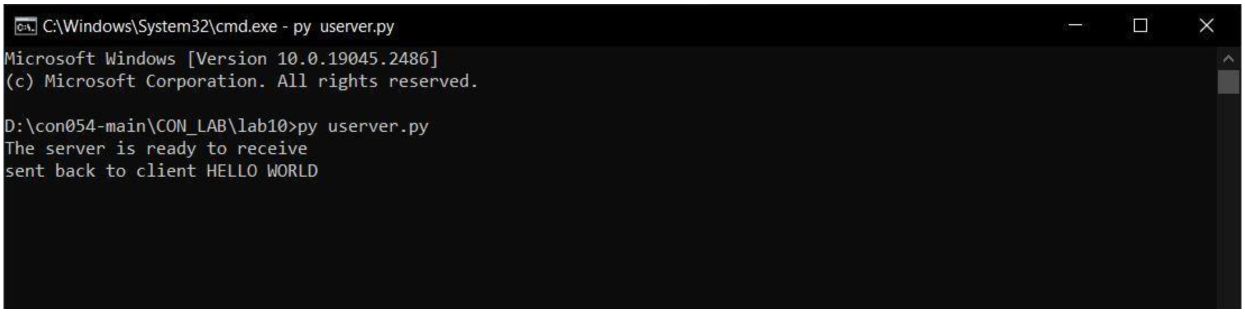
Output



```
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

D:\con054-main\CON_LAB\lab10>py userver.py
The server is ready to receive
```



```
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

D:\con054-main\CON_LAB\lab10>py uclient.py
Enter file name: try.txt
From Server: b'HELLO WORLD\n\n'

D:\con054-main\CON_LAB\lab10>
```



```
Microsoft Windows [Version 10.0.19045.2486]
(c) Microsoft Corporation. All rights reserved.

D:\con054-main\CON_LAB\lab10>py userver.py
The server is ready to receive
sent back to client HELLO WORLD
```