

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT

on

COMPUTER NETWORKS

Submitted by

LOKESH R (1BM20CS078)

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 “LAB COURSE **COMPUTER NETWORKS**” carried out by **LOKESH R (1BM20CS078)**, 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 2022-2023. 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. Seema Patil
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

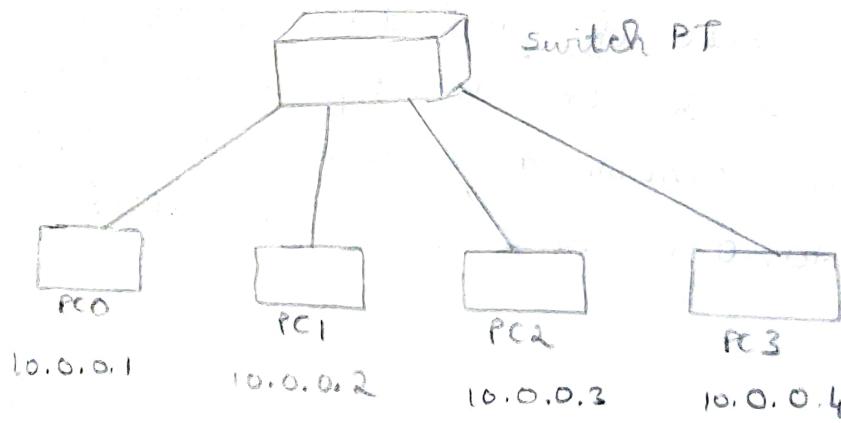
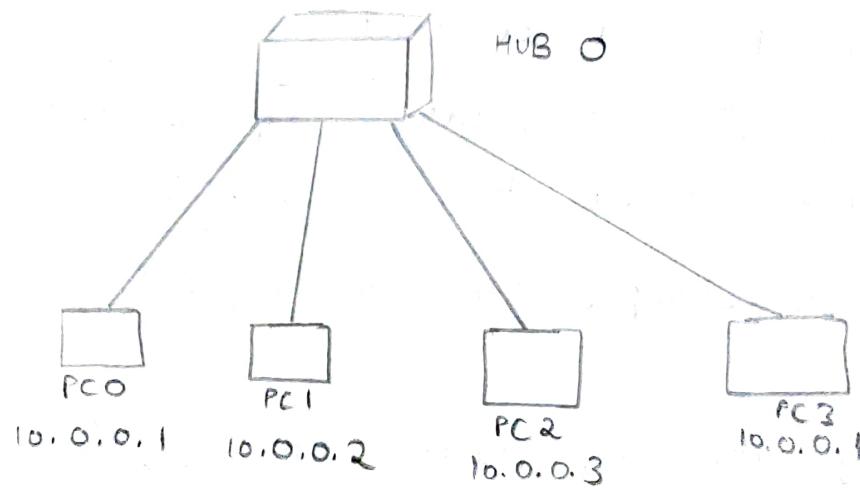
Index

Sl. No.	Date	Experiment Title	
1	10-11-2022	Creating a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.	
2	17-11-2022	Configuring IP address to Routers in Packet Tracer. Explore the following messages: Ping Responses, Destination unreachable, Request timed out, Reply	
3	24-11-2022	Configuring default route to the Router	
4	01-12-2022	Configuring DHCP within a LAN in a packet Tracer	
5	08-12-2022	Configuring RIP Routing Protocol in Routers	
6	15-12-2022	Demonstration of WEB server and DNS using Packet Tracer	
7	29-12-2022	Write a program for error detecting code using CRC-CCITT (16-bits).	
8	12-01-2023	Write a program for distance vector algorithm to find suitable path for transmission.	
9	12-01-2023	Implement Dijkstra's algorithm to compute the shortest path for a given topology.	
10	05-01-2023	Write a program for congestion control using Leaky bucket algorithm.	
11	28-01-2023	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.	
12	29-01-2023	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.	

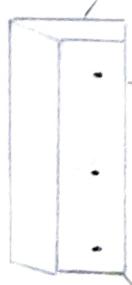
10/11/22

Experiment 1

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

Topology:-

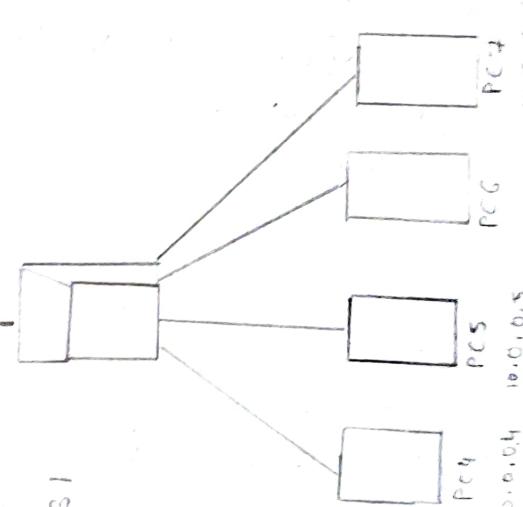
Switch



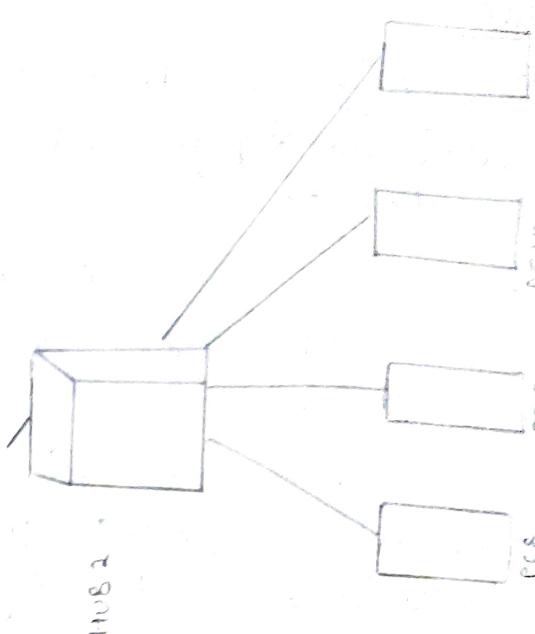
HUB 0



HUB 1



HUB 2



Hybrid

Procedure :-

* HUB

- Place 7 generic PC's and 1 generic hub in logical workspace and all 7 PC's are connected to hub, by copper straight wire.
- Set each PC's with IP addresses from 10.0.0.0 to 10.0.0.6 respectively and connect each PC to hub by copper straight wire.
- A simple PDU is placed on any 2 devices and message / packet passing can be seen in simulation mode , by clicking autocapture.
- In realtime mode a command prompt is opened for certain PC and following command is given to transfer message PING destination IP - addresses.

* Switch

- 4 generic PC's and one generic switch is placed on logical workspace.
- Set IP address for each PC from 10.0.0.7 to 10.0.0.10 and connect each PC to switch using copper straight wire.
- In simulation mode after placing simple PDU to any 2 PC's click auto - capture and packet transfer can be seen.
- In real time mode click on any PC and open command prompt and type 'PING dest. IP'



Hybrid :-

- 12 PC's , 3 hubs , 1 switch all generic are placed on to logical workspace.
- 3 generic hubs are connected to switch using copper cross over wire and 12 pc's are connected to 3 hubs , 4 PC each using copper straight wire after assigning IP address for each PC from 10.0.0.0 to 10.0.0.11 respectively.
- After selecting 2 PCs from different hubs with simple PDU's and clicking on auto-capture , packets passing simulation can be seen in simulation mode .
- In real time mode open command prompt by clicking any PC → devices → command prompt

Observations:-

* Hub

Learning outcome :-

- After source sends message to hub its broadcasted to all end devices but only destination device reads and send response back to hub for source to get response.
- Hub establishes connection to end-devices quickly and signals by green light.

Result:-

PING 10.0.0.3

PINGING 10.0.0.3 with 32 bytes of data
Reply from 10.0.0.3 with bytes = 32 time 0ms
PING Statistics for 10.0.0.3
Details of how many packets sent and received.

Learning observation :-

→ Unlike hub, switch does not give green signal immediately but takes some amount of time called learning time and the packet can be sent once green signal is generated.

Result :-

PING 10.0.0.5

PINGING 10.0.0.5 with 32 bytes of data

:

PING STATISTICS FOR 10.0.0.5

Hybrid :-

Learning outcomes :-

Message sent by one PC of one hub to switch is sent to destination hub which broadcast to all devices of that hub and only destined end devices sends back response to source of other hub.

Result :-

PING 10.0.0.4

PINGING 10.0.0.4 with 32 bytes of data

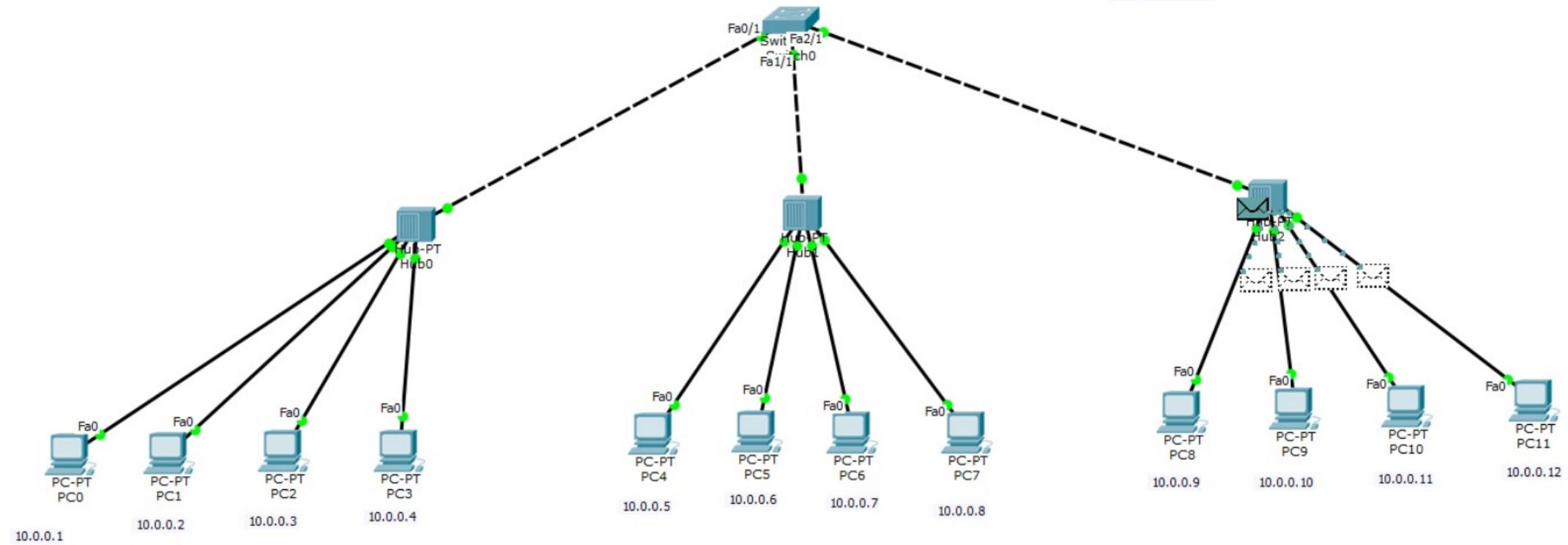
Reply from 10.0.0.4 : bytes = 32

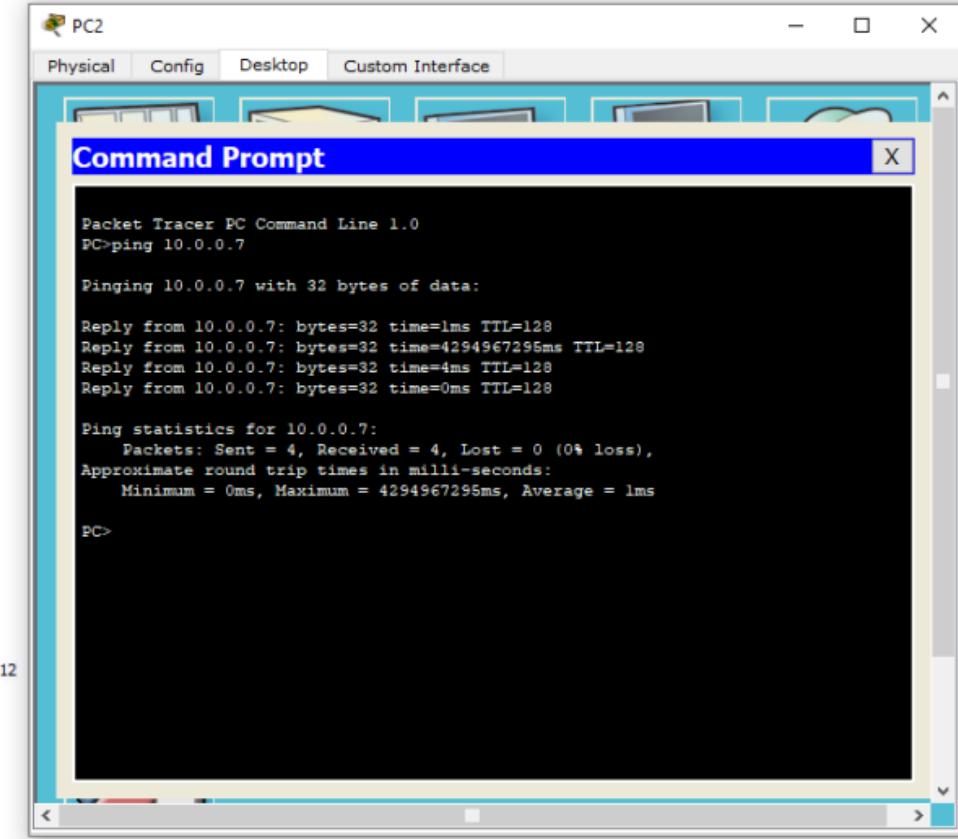
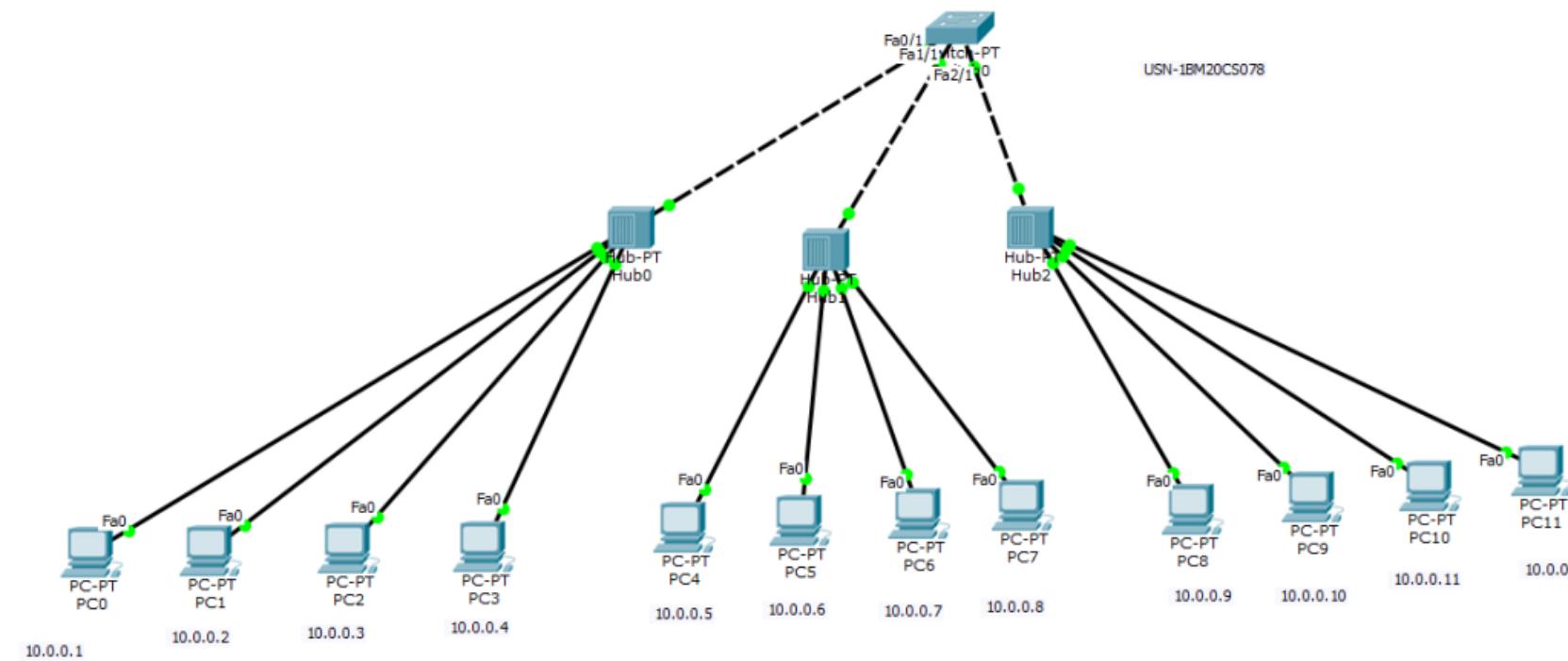
PING statistics for 10.0.0.4

"Details of number of packets sent and received"

elaborate
by

N
17/1/22



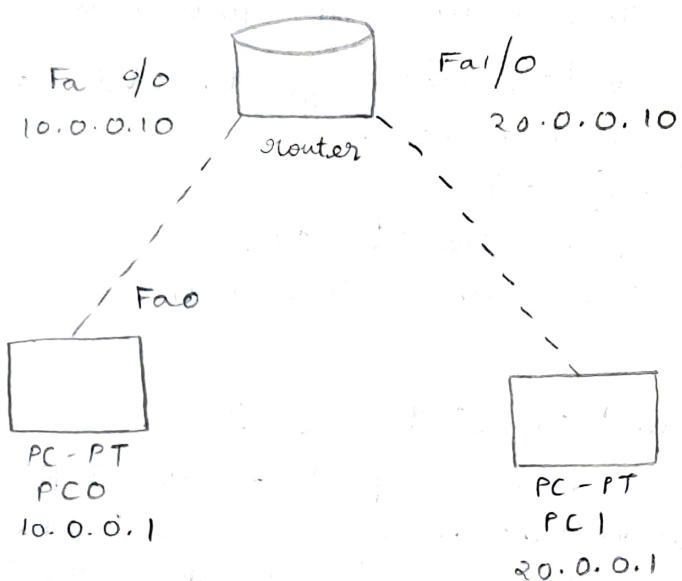


17/4/22

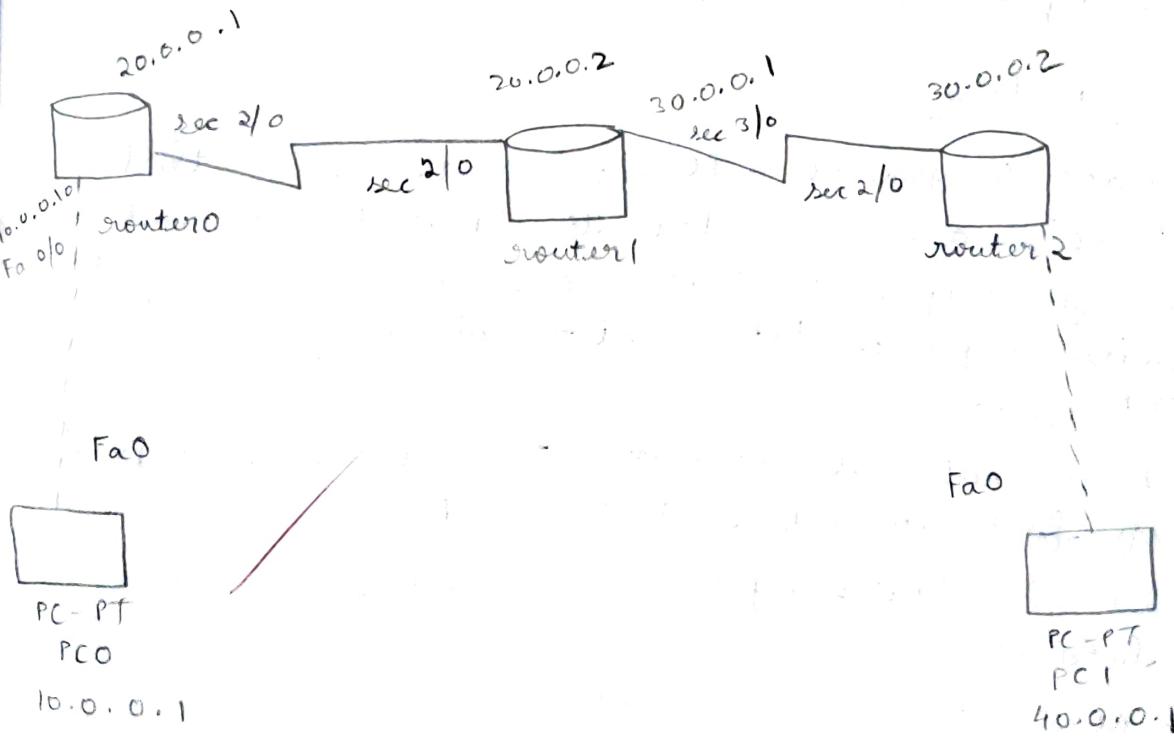
AIM:- Configuring IP address to routers in packet tracer. Explore the following messages:
PING response, destination unreachable, request timed out, reply.

Topology :-

One Router :-



Three Router :-



Procedure :-

One Router :-

- Place a generic PC, a generic router and notes with IP address and connect using copper cross over.
- Click on both PC's and set IP addresses as 10.0.0.1 and 10.0.0.2 with gateway for each as 10.0.0.10 and 10.0.0.20.
- Click on router, then go to CLI and execute following commands:
 - Enable → config - t
 - interface fast ethernet 0/0
 - ip address 10.0.0.10 255.0.0.0
 - no shut

Now connection between PC₀ and router turns green and repeat above steps for PCs until connection turns green. Route table can be seen by using "show ip route" command

- Now ping a PC by selecting one of the PC → desktop → command prompt.

Three Routers

- Place 2 generic PC's, 3 routers such that 2 PC's are connected to 1st and 3rd router using copper cross and 3 routers are connected with sequence 1st → 2nd → 3rd using serial DCE cable.
- Place notes to indicate IP addresses and ports of fast ethernet and serial ports
- Set IP address, gateway, subnet mask for both PC's.
- Now configure router 1
 - * Click → no
 - * Enable
 - * Config t

- * interface fastethernet 0/0
 - * ip address 10.0.0.10 255.0.0.0
 - * no shut
- Say this first PC and left half of router connection is established.
- * config t → interface serial 2/0 →
ip address 20.0.0.1 255.0.0.0 → no shut
- Now right side connection is established for 1st router.

→ do the same steps for router 2 and three with correct IP addresses and port selection to establish connection on both sides of each of the routers successfully.

→ After all connections green lights are shown as a result of successful completion, now if we ping we get "destination unreachable".

→ If we ping to the other router, we get "request timed out" as routers are not trained for indirect LAN's.

* Train router 1 by :-

ip route 30.0.0.0 255.0.0.0	20.0.0.2
ip route 40.0.0.0 255.0.0.0	20.0.0.2

* Train router 2 by :-

ip route 10.0.0.0 255.0.0.0	20.0.0.1
ip route 40.0.0.0 255.0.0.0	30.0.0.2

* Train router 3 by :-

ip route 10.0.0.0 255.0.0.0	30.0.0.1
ip route 20.0.0.0 255.0.0.0	30.0.0.1

→ Now pinging from PC0 to PC1 is successful.

Observations

One-router :-

Pinging output for first time

Ping 20.0.0.1

Pinging 20.0.0.1

Request timed out.

Reply from 20.0.0.1 : bytes = 32

Reply from 20.0.0.1 : bytes = 32

Reply from 20.0.0.1 : bytes = 32

Ping statistics for 20.0.0.1 : packets sent = 4,
received = 3, lost = 1

But when PC0 pings PC1 again or if reverse ping happens, we get reply all 4 times.

3 routers :-

- * Output when PC0 is pinged by PC1 or vice versa before routers are trained of unknown IP's

Ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data

Reply from 10.0.0.10 : destination host unreachable.

PING statistics 40.0.0.1

Packets sent = 4, received = 0, lost = 4

* PING 20.0.0.2

Request timed out

Request timed out

Request timed out

Request timed out

Pinging statistics 20.0.0.2

Packets sent = 4, received = 0, lost = 4

* When routers are trained

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 = 2 ms

Reply from 40.0.0.1 : bytes = 32 time = 2 ms

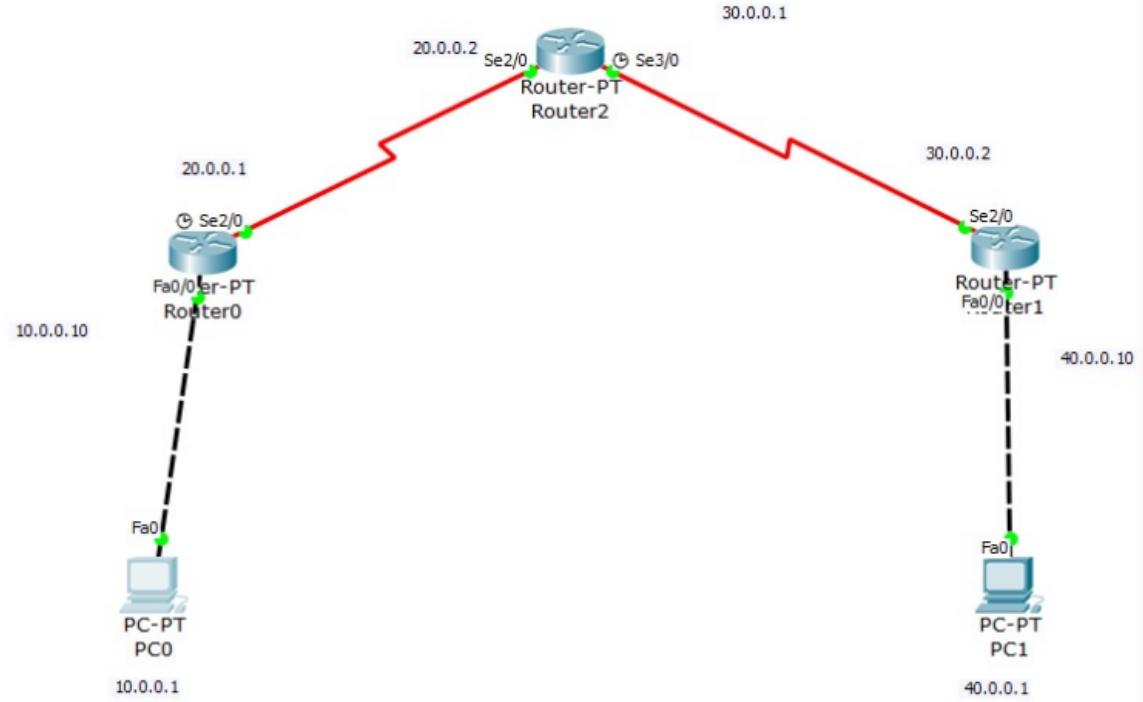
Reply from 40.0.0.1 : bytes = 32 time = 2 ms

Pinging statistics for 40.0.0.1 packets sent = 4,

Received = 3

N
24/11/22

IBM20CS078



PC0

Physical Config Desktop Custom Interface

Command Prompt

```
Packet Tracer PC Command Line 1.0
PC>ping 40.0.0.1

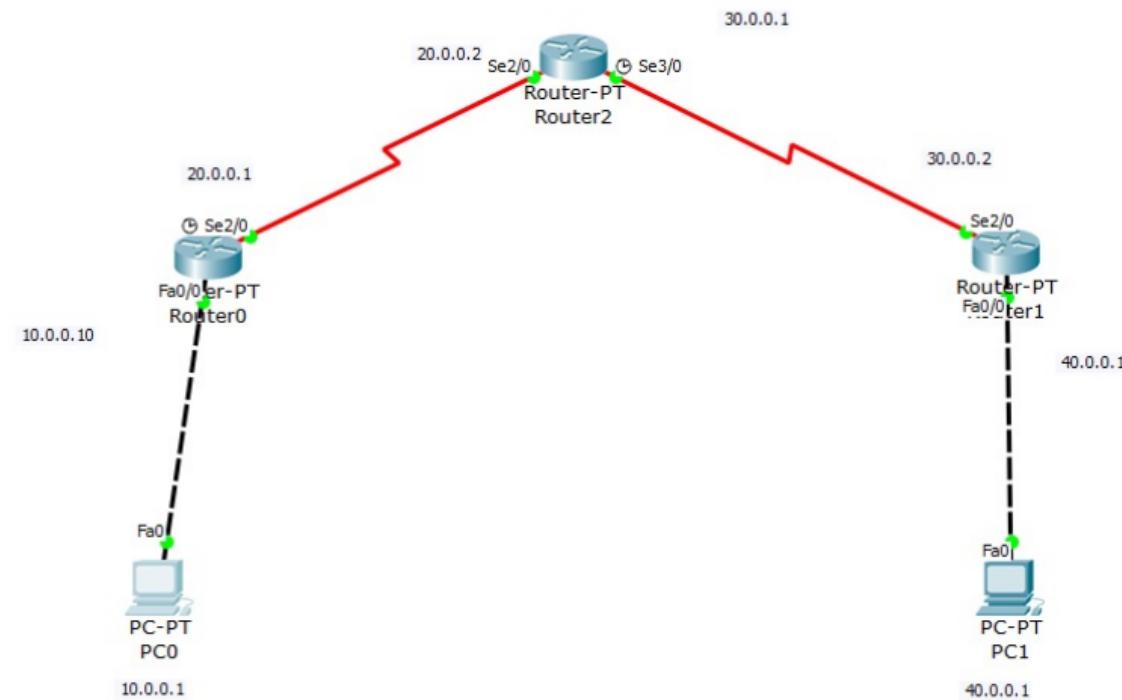
Pinging 40.0.0.1 with 32 bytes of data:

Reply from 10.0.0.10: Destination host unreachable.
Reply from 10.0.0.10: Destination host unreachable.
Request timed out.
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),

PC>
```

1BM2CS078



PC0

Physical Config Desktop Custom Interface

Command Prompt

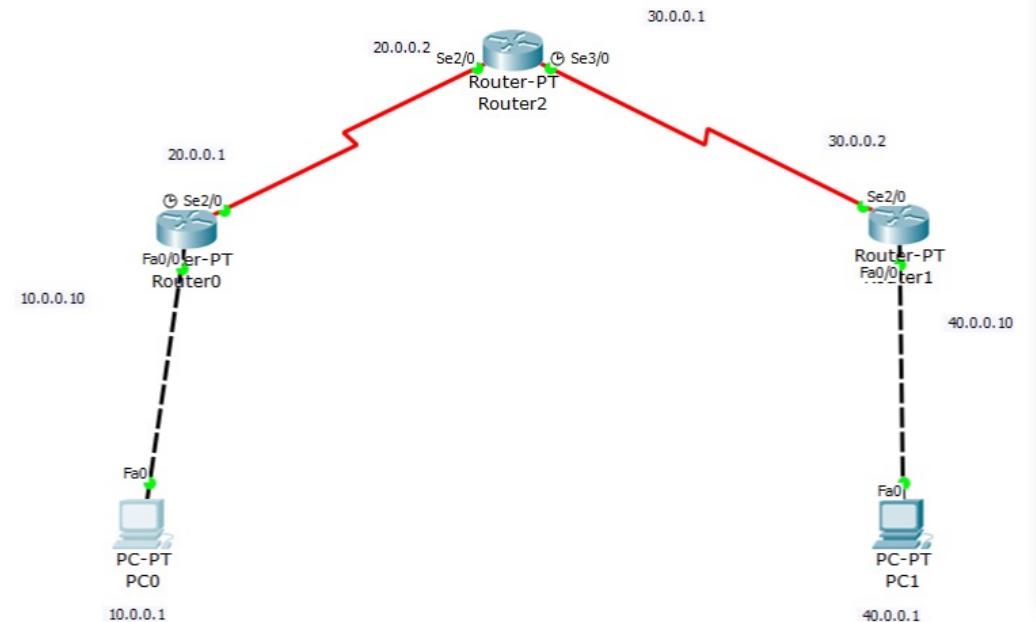
```
PC>ping 40.0.0.1

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=8ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms 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 = 2ms, Maximum = 10ms, Average = 6ms

PC>
```



PC0

Physical Config Desktop Custom Interface

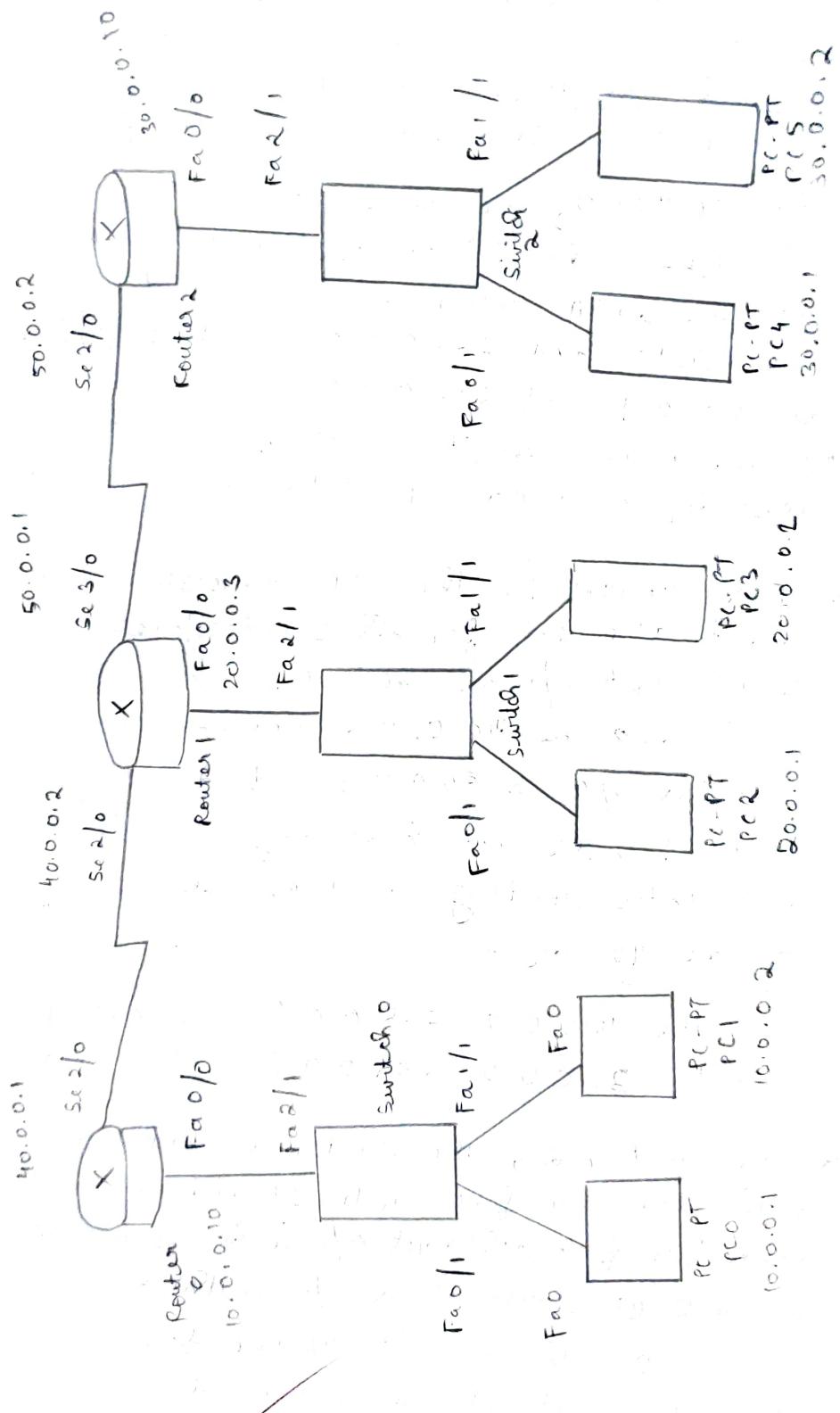
Command Prompt

```
PC>ping 20.0.0.2
Pinging 20.0.0.2 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 20.0.0.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
PC>
```

Aim :- Configuring default route to the router.

Topology :-



Procedure:-

→ Place 6 generic PC's, 3 switches and 3 routers and connect 2 PC's to each switch with copper straight wire and each switch is connected to one router with a copper straight wire and 3 routers are connected among themselves by serial DCE cable.

PC0 and PC1 gateway - 10.0.0.1
PC2 and PC3 gateway - 20.0.0.1
PC4 and PC5 gateway - 30.0.0.1

→ A PC is clicked to set the attributes for a PC and each PC has 3 attributes which are the IP address, subnet mask and the gateway and all the three are set according to the nodes placed. This process is done for all 6 PCs.

→ For router 0, the configuration are done in the CLI. The IP address and subnet mask are set for both interface fastethernet 0/0 as 10.0.0.10 and 255.0.0.0 and serial 2/0 as 40.0.0.1 and 255.0.0.0.

Router 1 is default router for Router 0 and done by command ip route 0.0.0.0 0.0.0.0 40.0.0.1

→ For router 1, the IP addresses and subnet mask are set for all 3 interfaces, fastethernet 0/0 as 20.0.0.3 and 255.0.0.0 serial 2/0 as 40.0.0.2 and 255.0.0.0 and serial 3/0 as 50.0.0.1 and 255.0.0.0.

Router 1 does not have any default routers and static routing is done for network 10 and 40 by :-

~~ip route 10.0.0.0 255.0.0.0 40.0.0.1~~
~~ip route 30.0.0.0 255.0.0.0 50.0.0.2~~

→ For router 2, the router is configured in both the interface with IP address and subnet mask as fast ethernet 0/0 with 30.0.0.10 and 255.0.0.0 and serial 2/0 with 50.0.0.2 and 255.0.0.0. The default router for router 2 is router 1 and set by command :-

ip route 0.0.0.0 0.0.0.0 50.0.0.1

Ping command is executed from 10.0.0.1 to 20.0.0.1
and from 10.0.0.1 to 30.0.0.2

Observations:-

Learning outcomes:-

- One router cannot have 2 default routers.
- The default router for first router is the middle router because any packets which have to delivered will go to the middle router.
- The default router for third router is the middle router for same reason.
- The middle router does not have any default router because if one of the router is made default then there is a chance that the packets which are to be sent to the switch and sent to the router.

Result :-

ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data

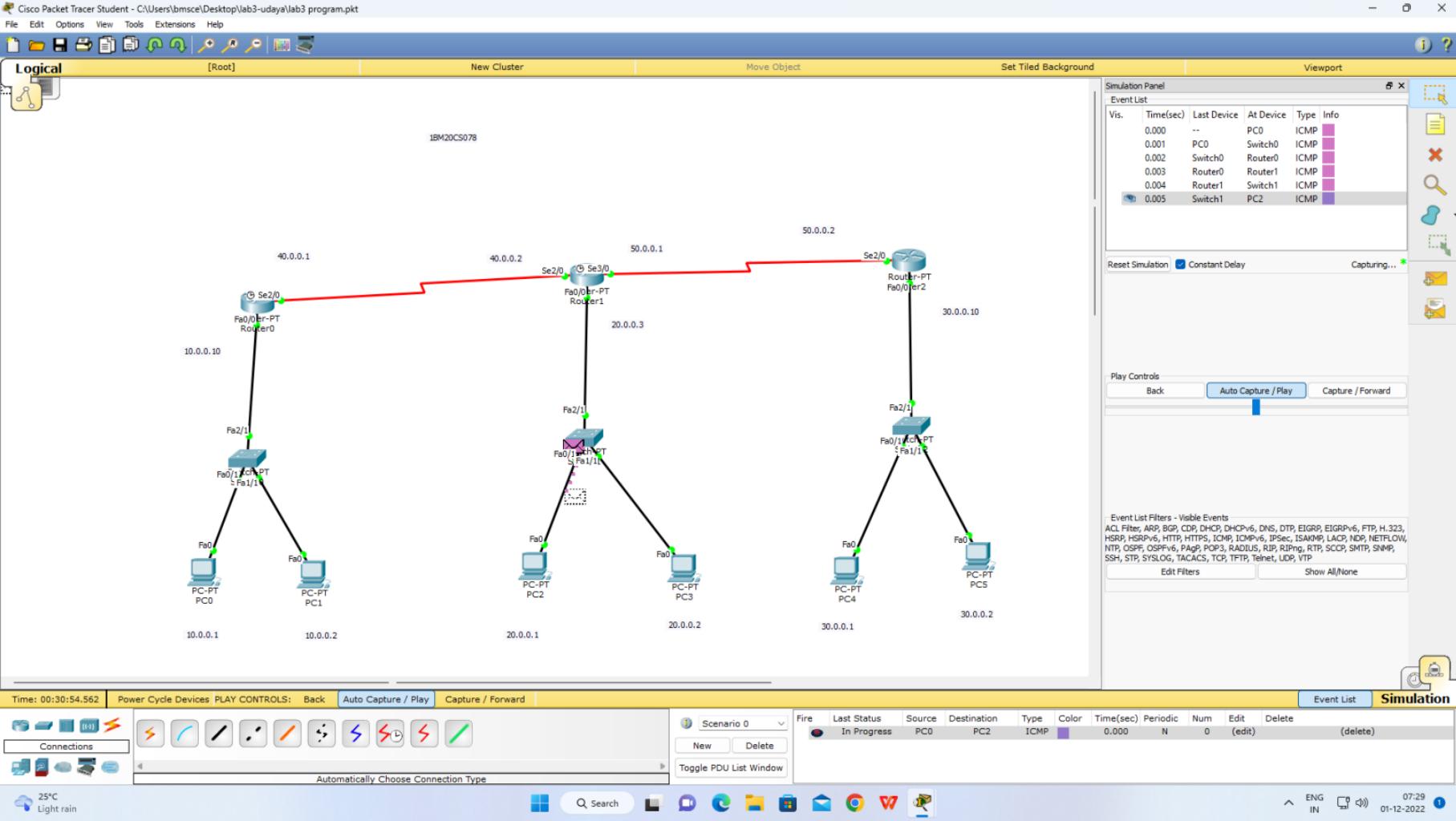
Request timed out

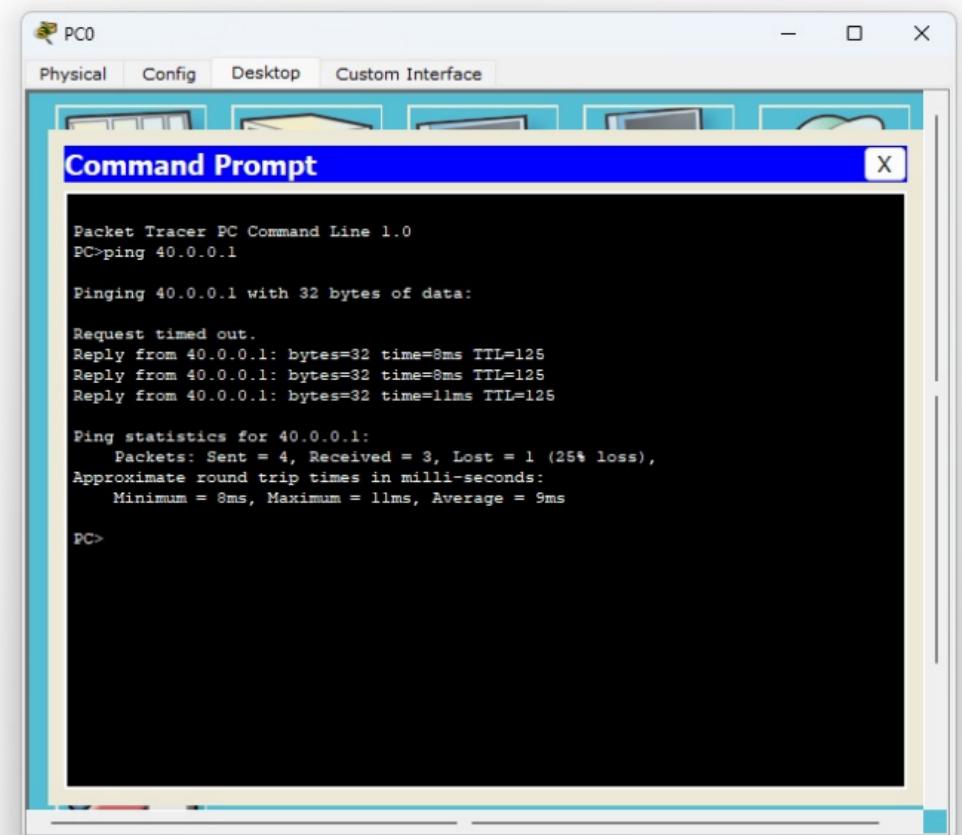
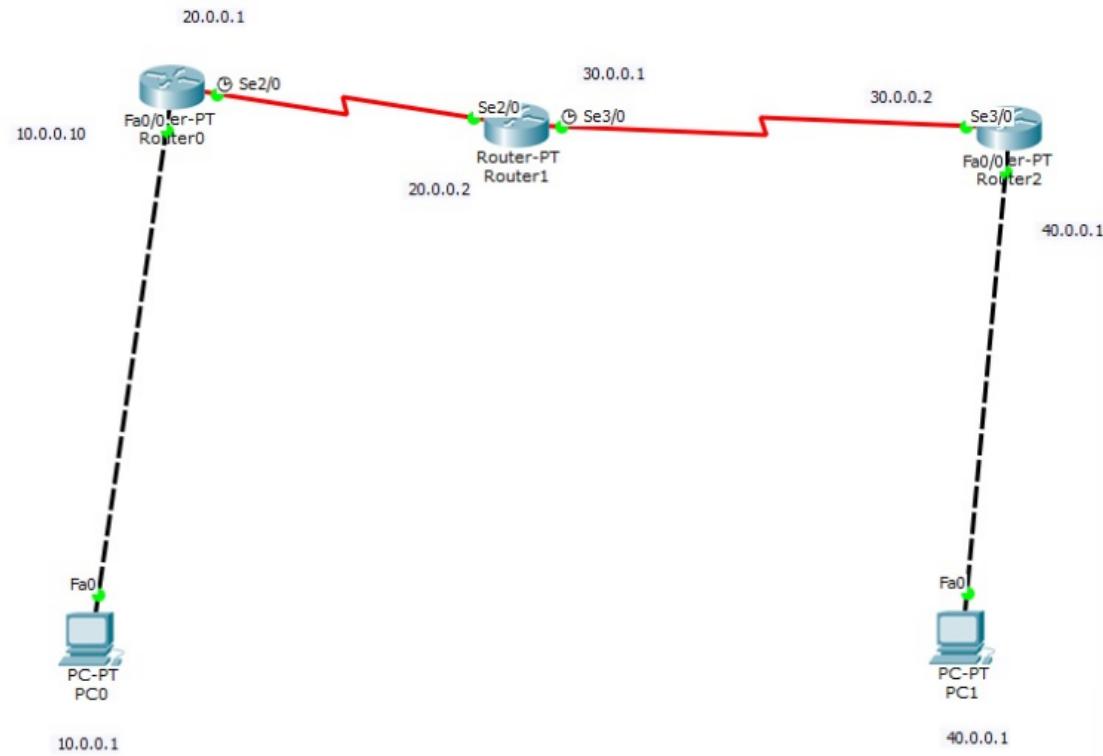
Reply from 20.0.0.1 bytes = 32 , time = 1 ms , TTL = 126

Reply from 20.0.0.1 bytes = 32 , time = 2 ms , TTL = 126

Reply from 20.0.0.1 bytes = 32 , time = 0 ms , TTL = 126

N
24/11/22





Logical

[Root]

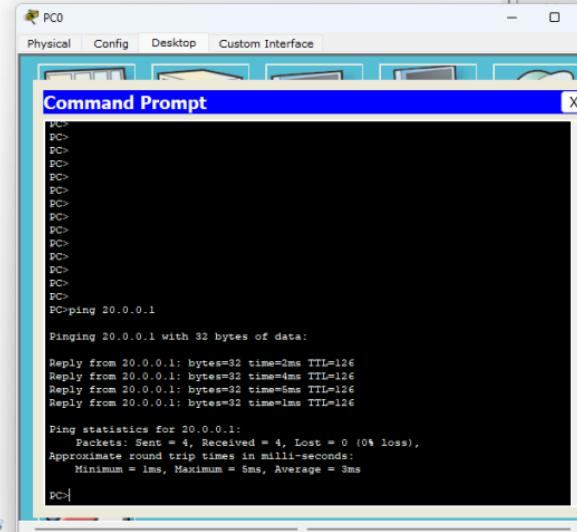
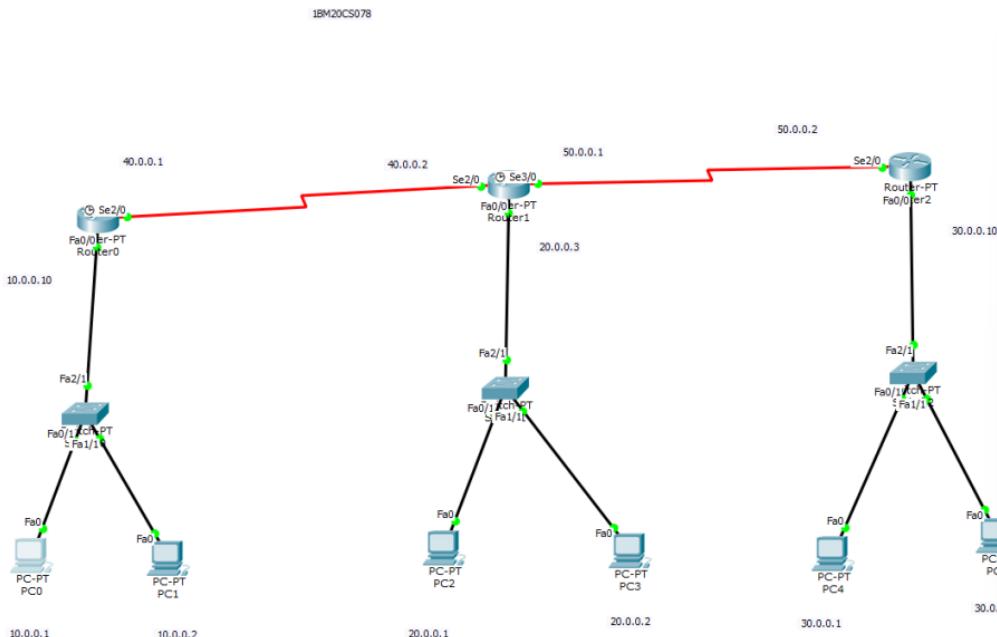
New Cluster

Move Object

Set Tiled Background

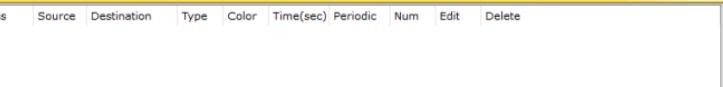
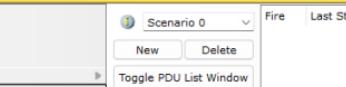
Viewport

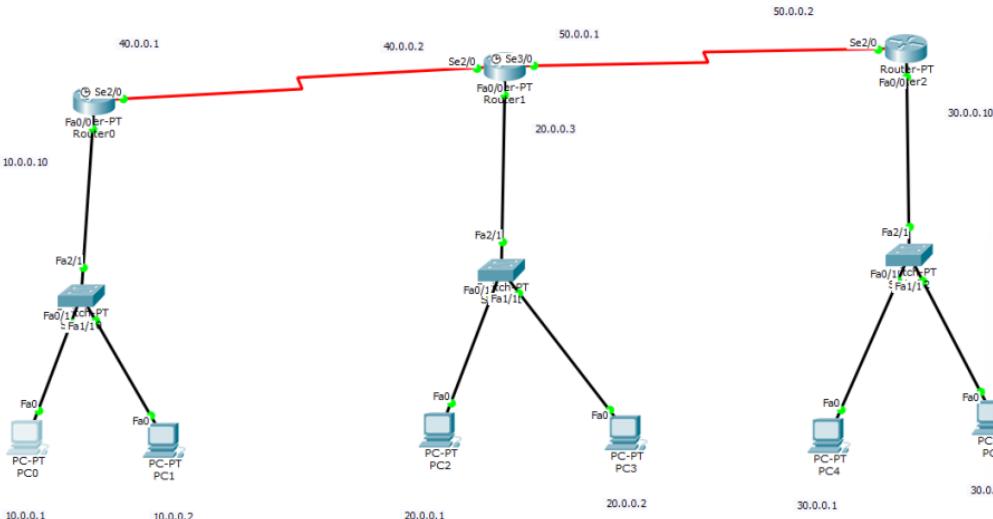
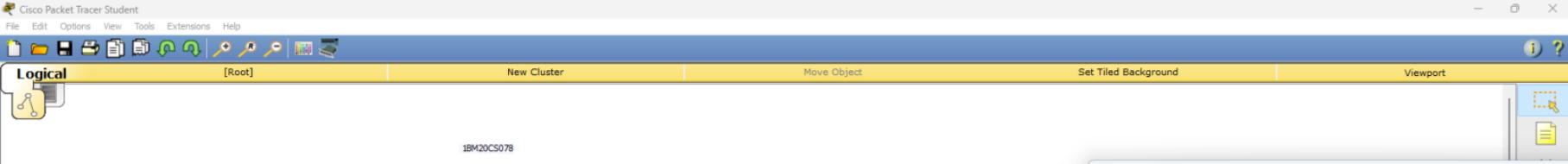
i ?



Realtime

Time: 00:22:18 Power Cycle Devices, Fast Forward Time



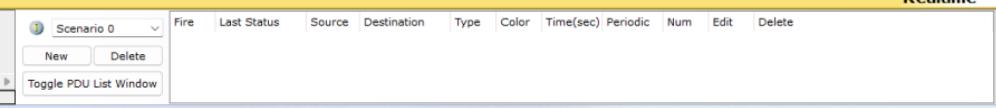


```
PC>ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:
Reply from 20.0.0.1: bytes=32 time=2ms TTL=126
Reply from 20.0.0.1: bytes=32 time=4ms TTL=126
Reply from 20.0.0.1: bytes=32 time=5ms TTL=126
Reply from 20.0.0.1: bytes=32 time=1ms TTL=126

Ping statistics for 20.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 6ms, Average = 3ms

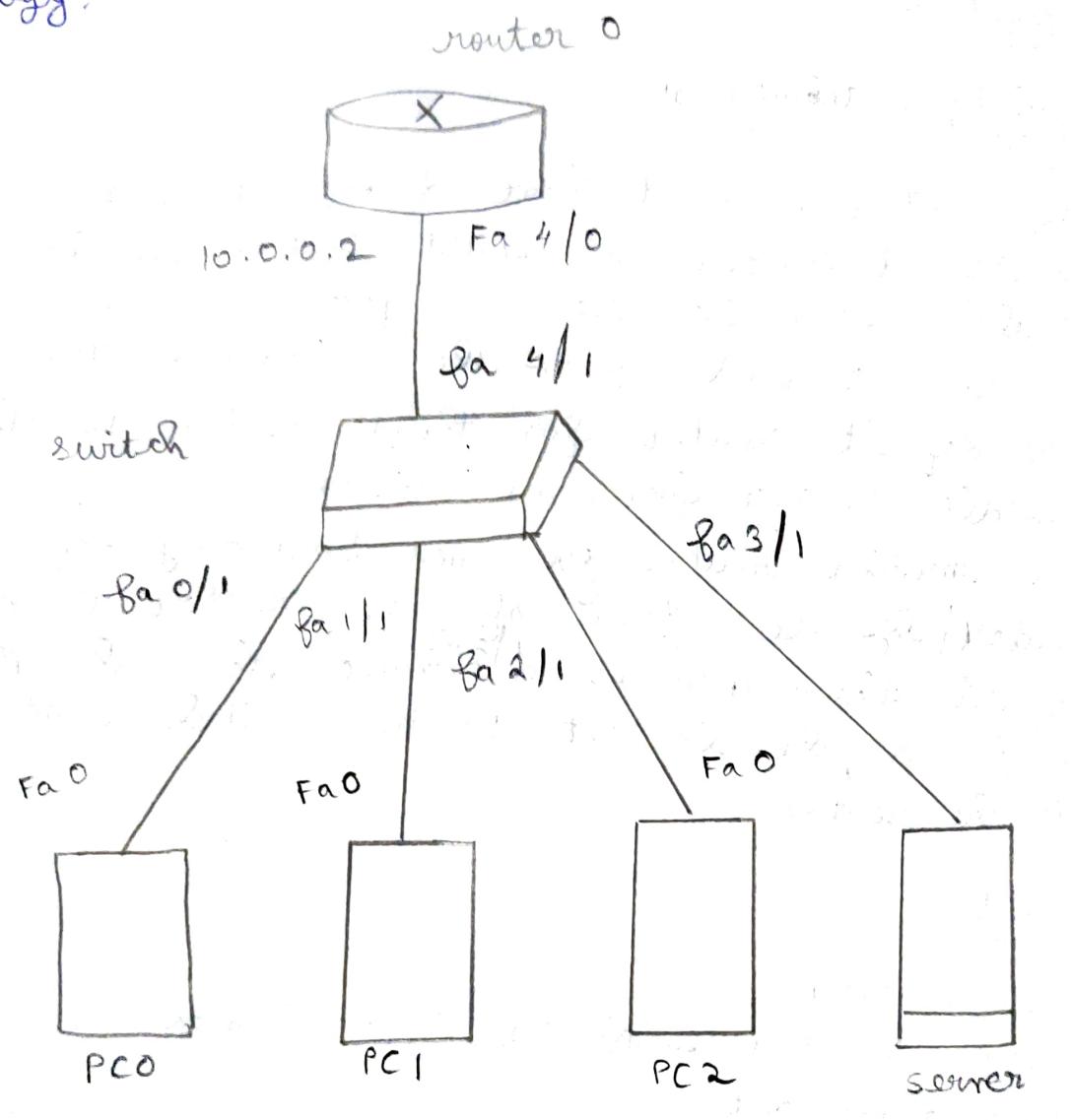
PC>ping 30.0.0.1
Pinging 30.0.0.1 with 32 bytes of data:
Reply from 30.0.0.1: bytes=32 time=2ms TTL=125
Reply from 30.0.0.1: bytes=32 time=12ms TTL=125
Reply from 30.0.0.1: bytes=32 time=12ms TTL=125
Reply from 30.0.0.1: bytes=32 time=3ms TTL=125

Ping statistics for 30.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 2ms, Maximum = 21ms, Average = 9ms
```



AIM :- Configuring DHCP within a LAN in a packet tracer.

Topology :-



Procedure:-

- * Place 3 PC's, one router, one generic switch, one server and connect everything with copper straight wire.
- * Place note 10.0.0.2 for router indicating gateway and 10.0.0.1 for server indicating its IP address
- * Interface the corresponding fastethernet interface for router using CLI :-
 - enable
 - config t
 - interface fastethernet 3/0
 - ip address 10.0.0.2 255.0.0.0
 - no shut
- * Now configure server by clicking services tab in servers
 - set default gateway
 - set DNS server
 - set starting IP address as 10.0.0.3
 - set max. no of users as 8
 - set TFTP server as 10.0.0.1
 - Then click on save.
- * Then click on PC0, go to desktop, click on IP configuration and you click on DHCP.

Observations:-

When we select DHCP mode, it automatically fetches, gateway from server for which we had configured already.

Result :-

ping 10.0.0.5

pinging 10.0.0.5 with 32 bytes of data :-

Reply from 10.0.0.5 , bytes = 32 time = 1ms TTL = 128

Reply from 10.0.0.5 , bytes = 32 time = 0ms TTL = 08

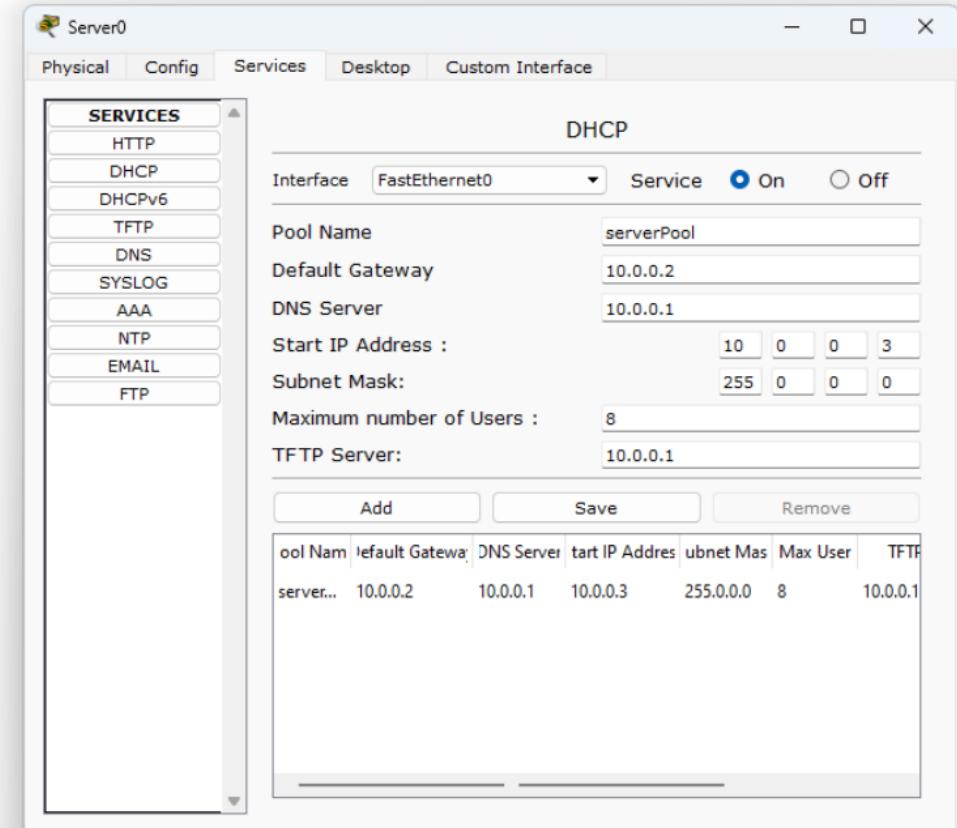
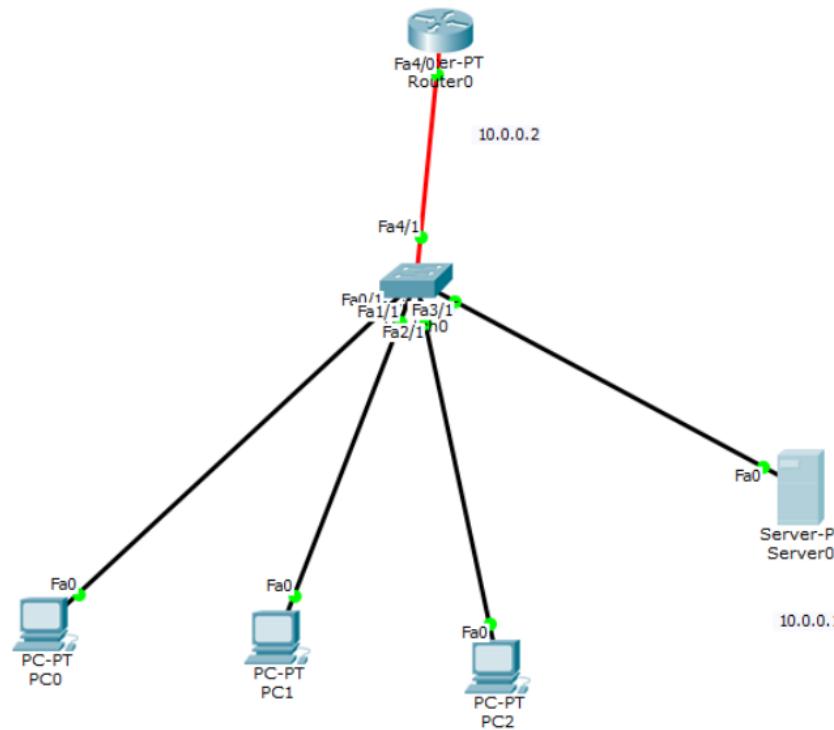
Reply from 10.0.0.5 , bytes = 32 time = 0ms TTL = 128

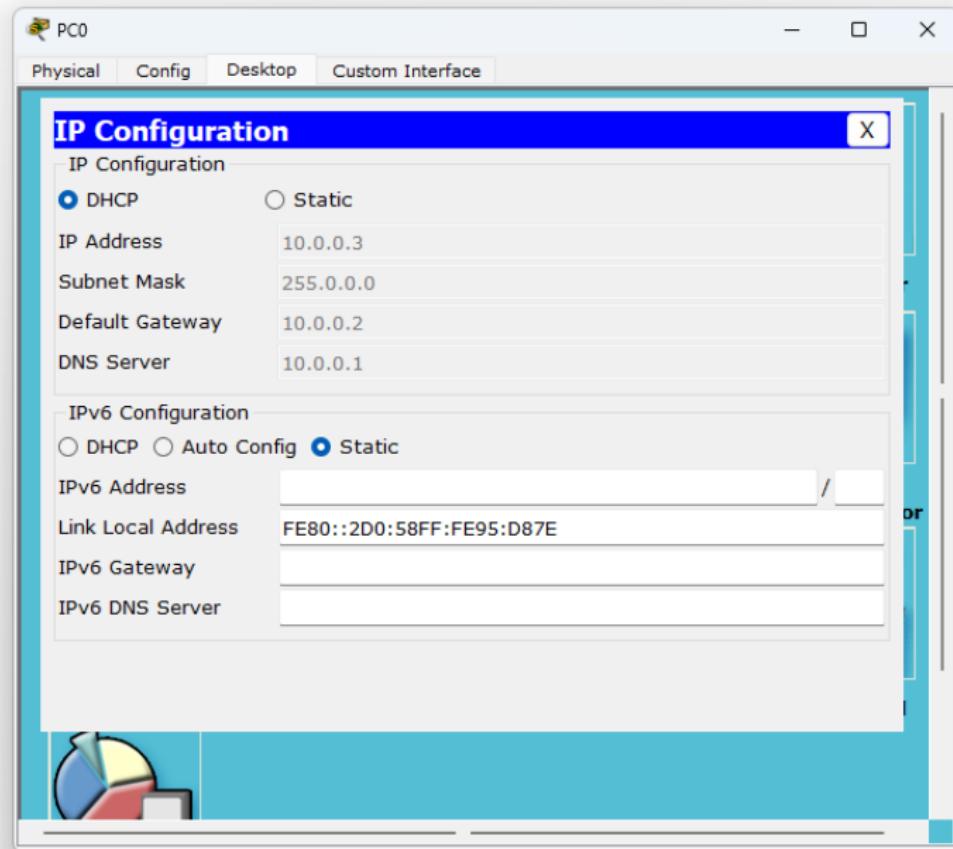
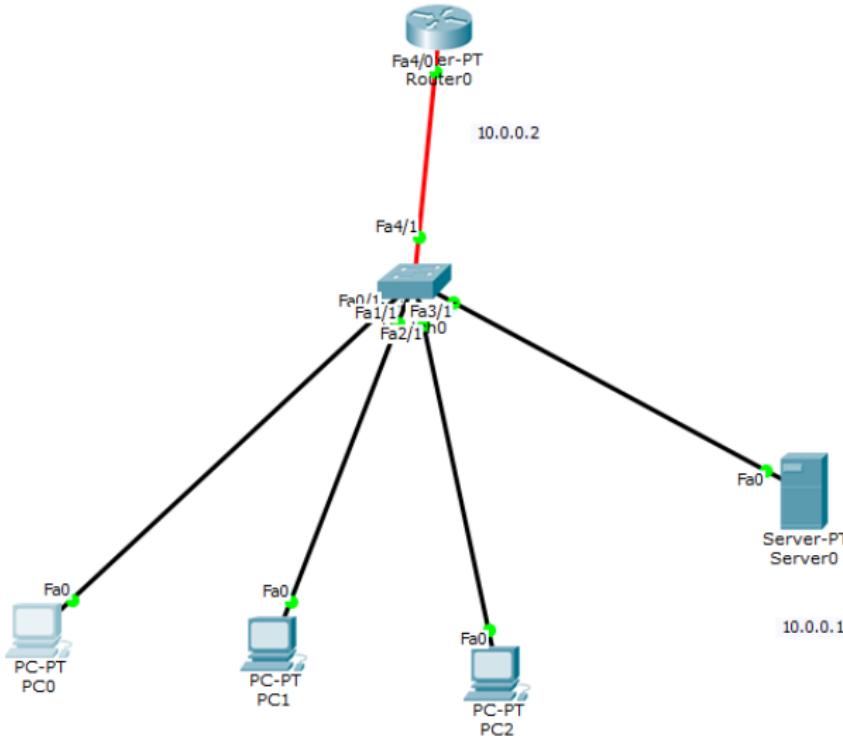
Reply from 10.0.0.5 , bytes = 32 time = 0ms TTL = 128

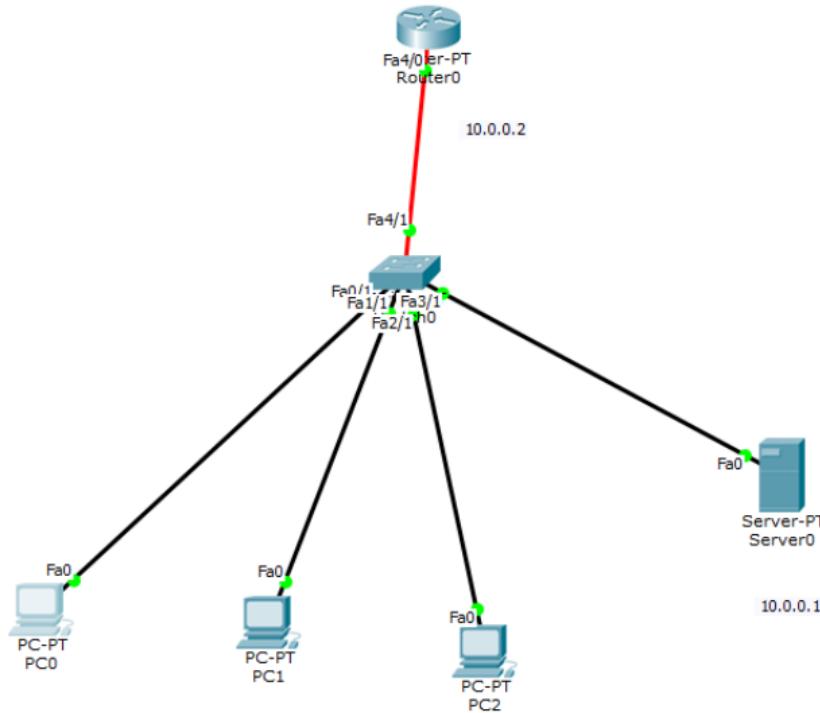
ping statistics for 10.0.0.5 :-

Packets sent = 4 , Received = 4 , Loss = 0 (0% loss)

✓
8/12/22







A screenshot of a Windows-style application window titled "PC0". The window contains a "Command Prompt" window with the following text:

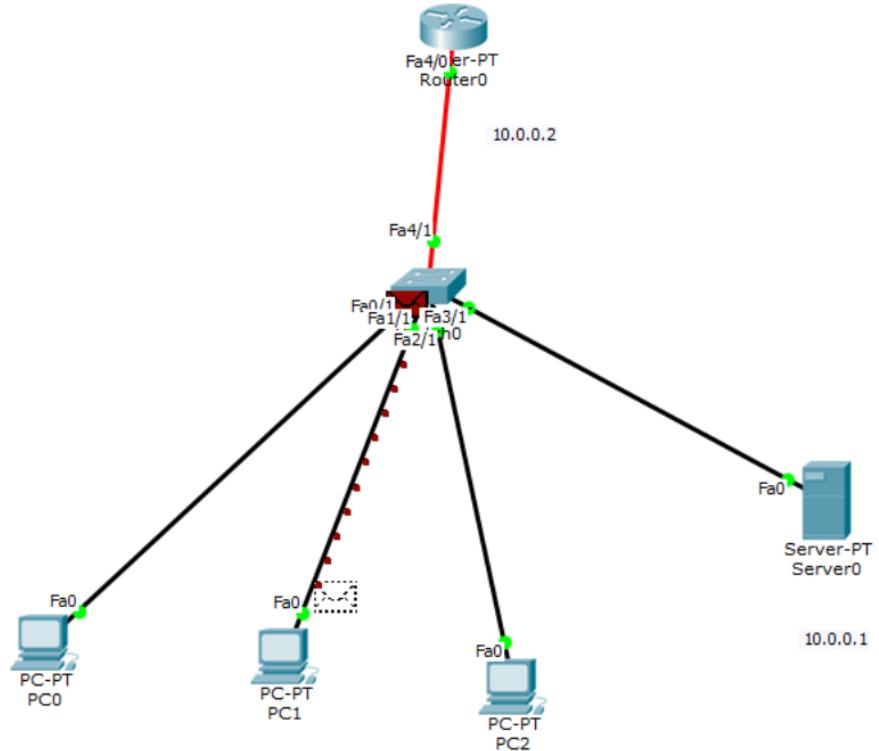
```
Packet Tracer PC Command Line 1.0
PC>ping 10.0.0.5

Pinging 10.0.0.5 with 32 bytes of data:

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

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

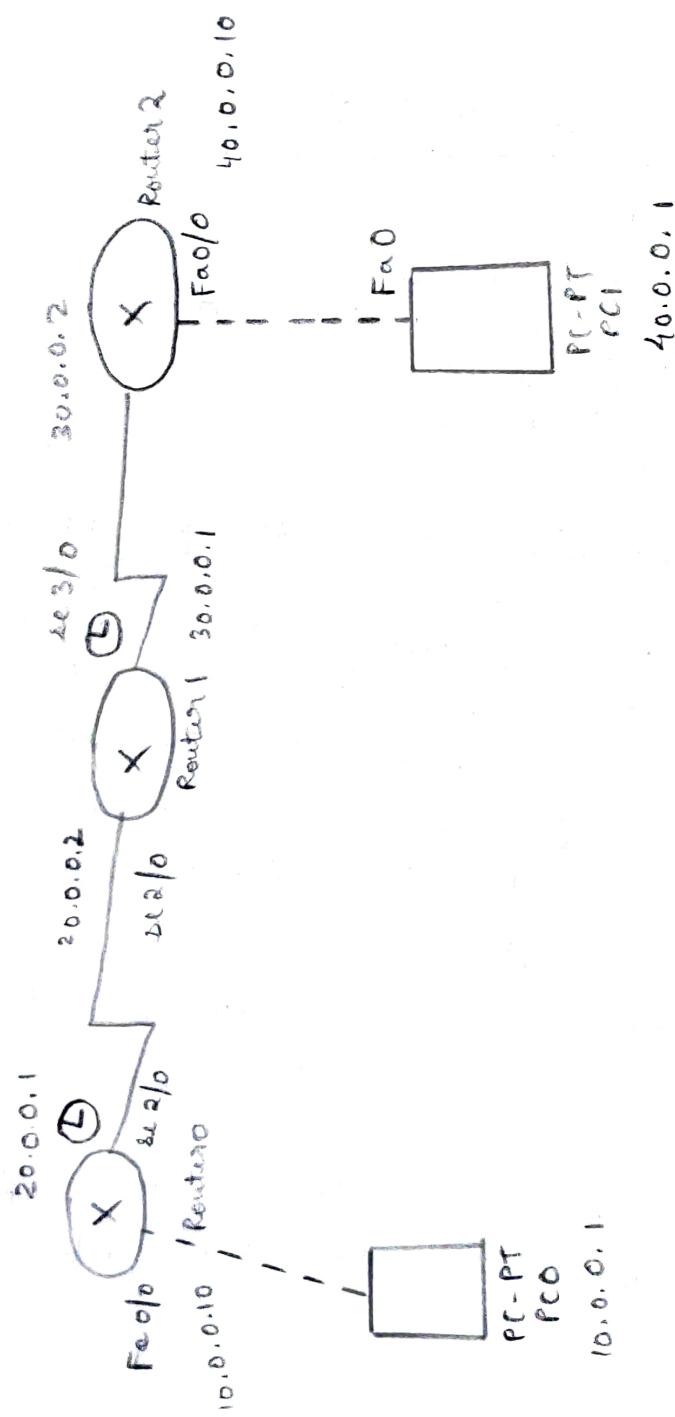
PC>|
```



8/12/22

AIM :- Configuring RIP routing protocol in routers

RIP :- RIP is routing information protocol which finds best path between source and destination network. It is a distance vector routing protocol.



Procedure :-

- * Place 2 generic PC's, 3 routers, and place notes to indicate respective IP addresses
- * For PC-PT PC0 set IP address 10.0.0.1
subnet 255.0.0.0
and gateway as 10.0.0.10
- * For PC-PT PC1 set IP address 40.0.0.1
subnet 255.0.0.0
and gateway as 40.0.0.10

* For router 0

→ enable
→ config t
→ interface fastethernet 0/0
→ ip address 10.0.0.10 255.0.0.0
→ no shut

⇒ interface serial 2/0
→ ip address 20.0.0.1 255.0.0.0
→ encapsulation ppp
→ clock rate 64000
→ no shut

- * use the above commands for interfacing router which has clock symbol in cable near it and for interfacing other router use the same commands except "clock rate 64000"
- * Place correct values while interfacing the routers.
- * Once all the green lights are visible, follow the commands below for the routers:
 - router rip
 - network 10.0.0.0 } known networks
 - network 20.0.0.0 } are written here.
 - exit
- * Repeat above commands for router 1 and router 2 with respective network addresses.

Observation :-

- * In static IP routing we need to teach the routers independently, but we make use of RIP so that routing becomes easy when large number of routers are used.

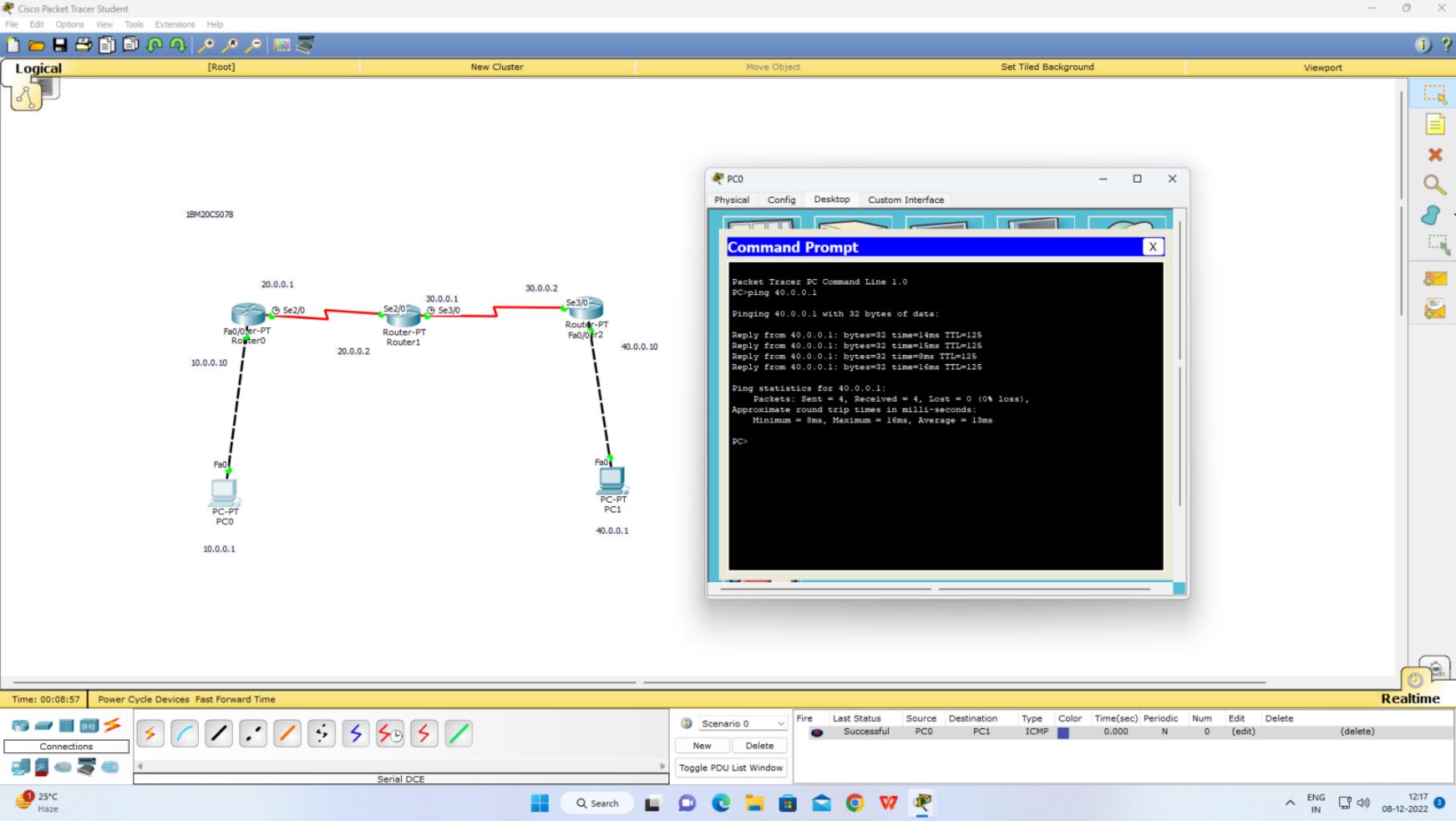
DCE - ?

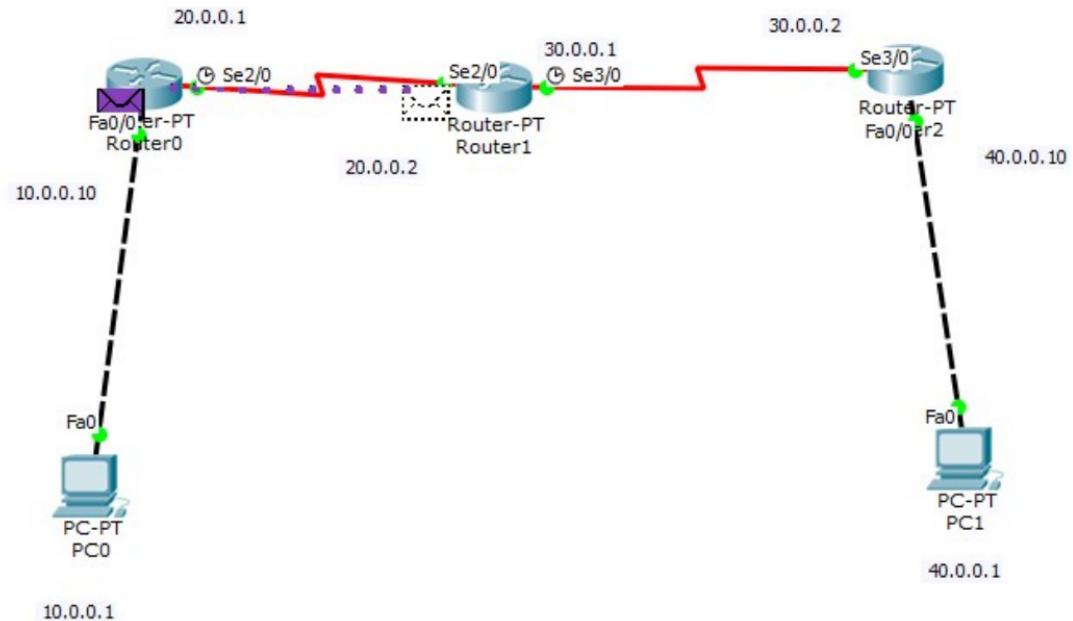
Result :-

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1	bytes = 32	time = 14ms	TTL = 125
Reply from 40.0.0.1	bytes = 32	time = 15ms	TTL = 125
Reply from 40.0.0.1	bytes = 32	time = 0ms	TTL = 125
Reply from 40.0.0.1	bytes = 32	time = 16ms	TTL = 125

✓ ping statistics for 40.0.0.1
packets sent = 4 > Received = 4 > lost = 0.

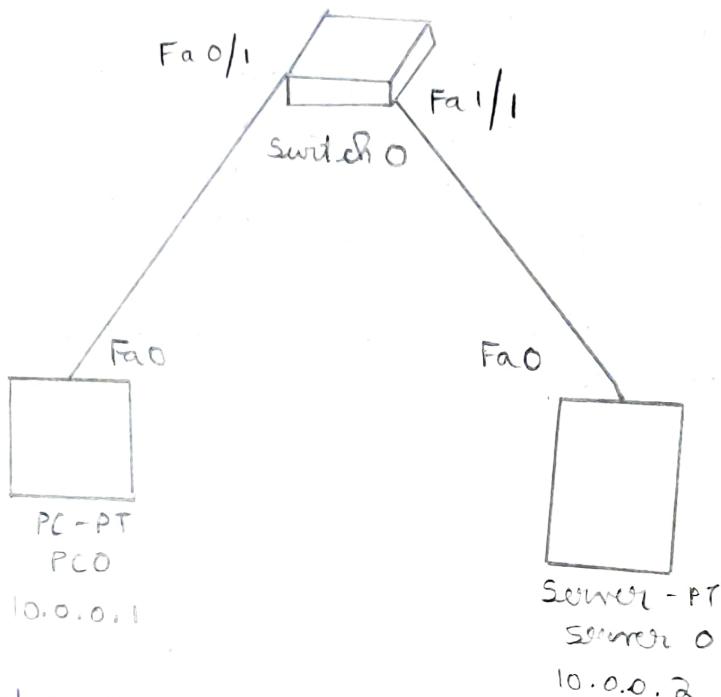




15/12/22

AIM:- Demonstration of WEB Server and D_P using packet tracer.

Topology:-



Procedure :-

- * Set the IP address of PC and server according to the figure. From the PC invoke the web browser in the desktop tab and give IP of server.
- * Now go to services tab in properties of server, click on HTTP. A window is seen. Here click on edit button of index.html and change text from CISCO to BMSC. Save to overwrite and see modified page.
- * Now to activate DNS, click on DNS services. Select an, enter the name and IP as seen in the figure. Click on add button. IP \Rightarrow 10.0.0.2
- * Now give name in the web browser of PC to check if it is displaying index.html.

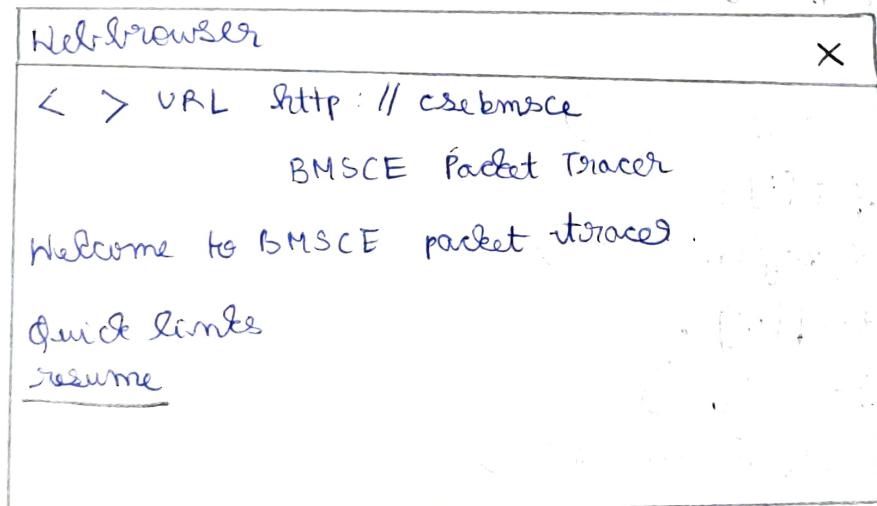
Custom - page

- Create a new page resume.html and save it in http services.
- Change hyperlink in index.html to link the above created file.
- Check the output in web browser of PC by click on hyper-link.

Observations

- We can view the webpage then we type "csebmse" in browser because 10.0.0.2 address is mapped to the name "csebmse" by domain name system concept.
- Mapping is required its difficult to users to remember IP's, hence IP is mapped with a name.

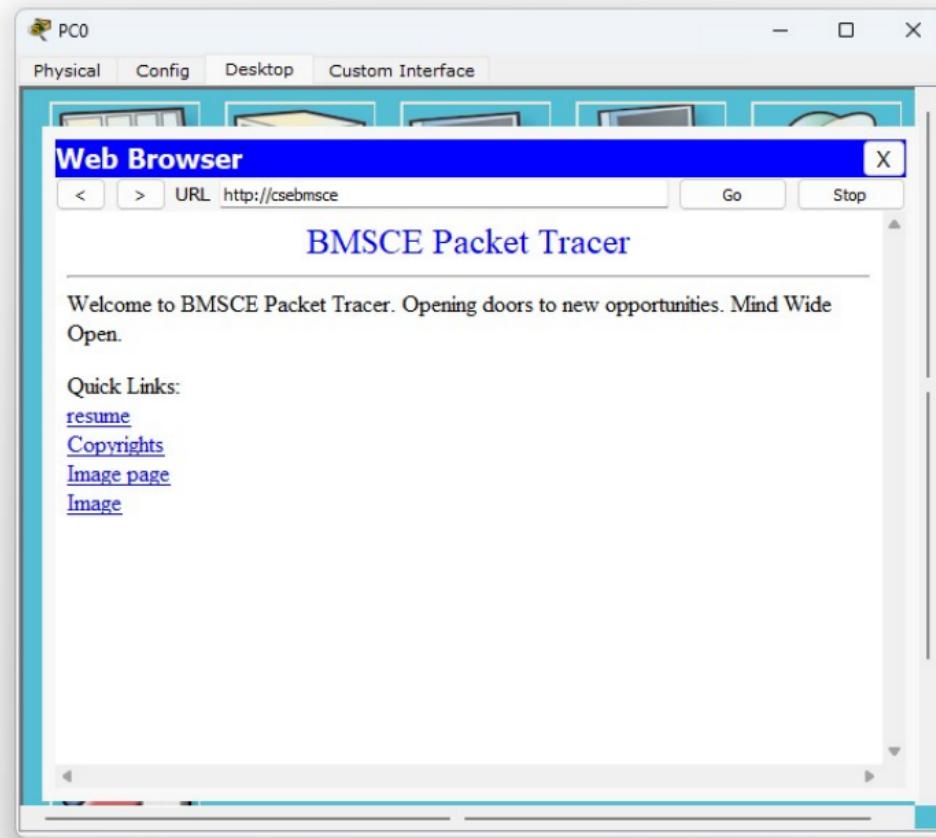
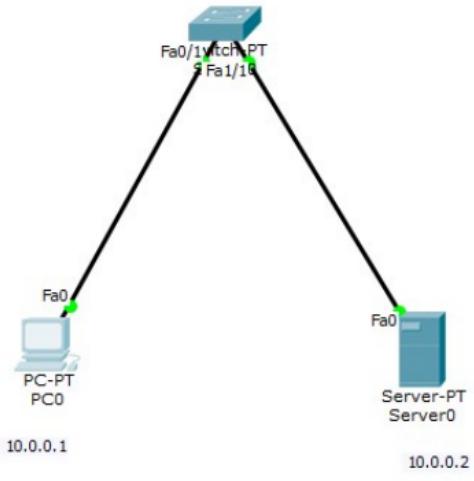
Result :-

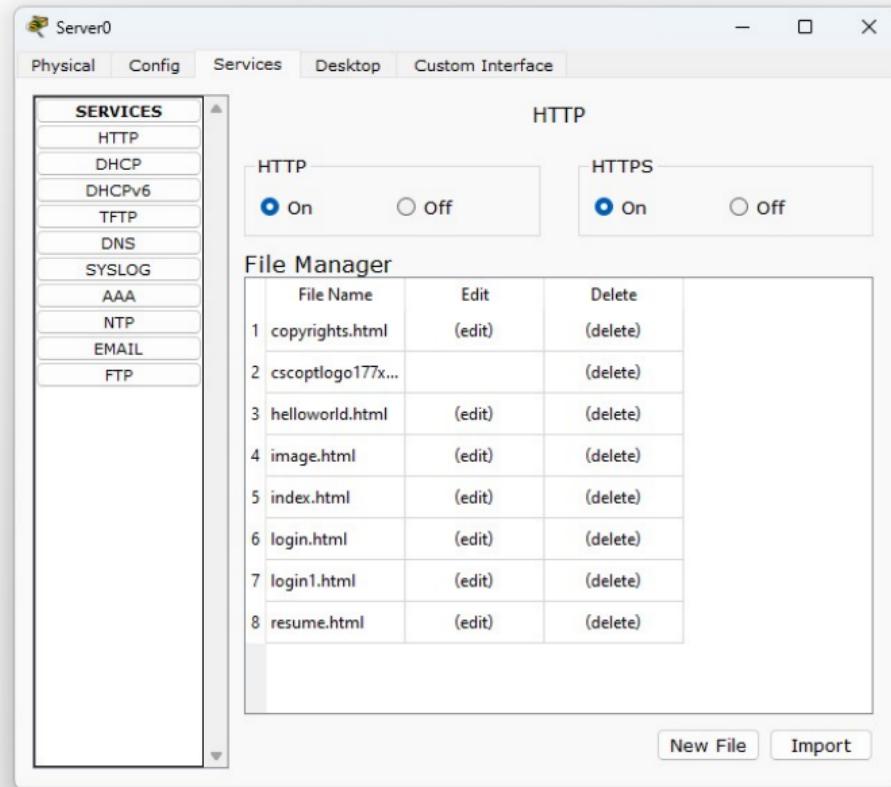
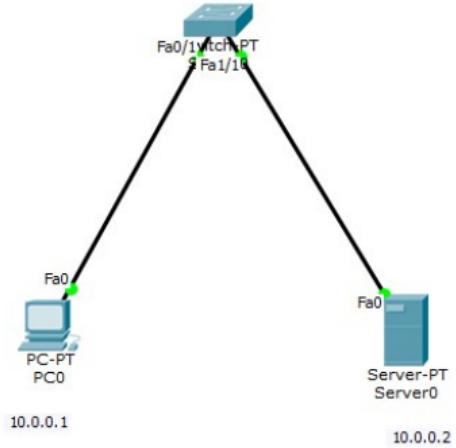


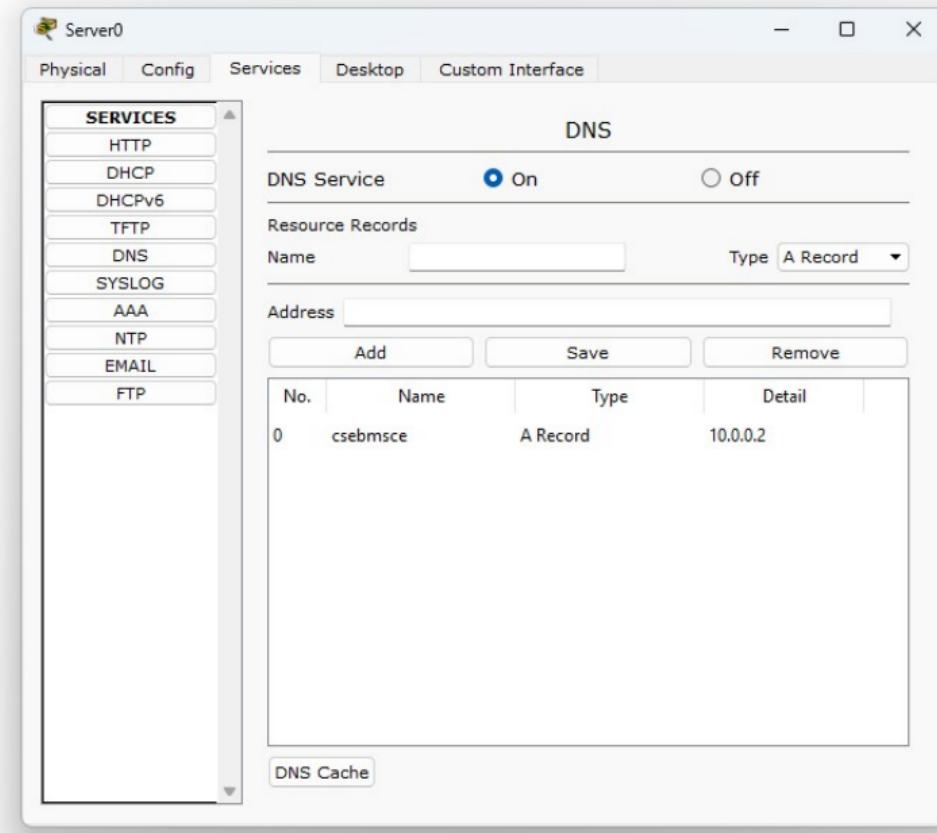
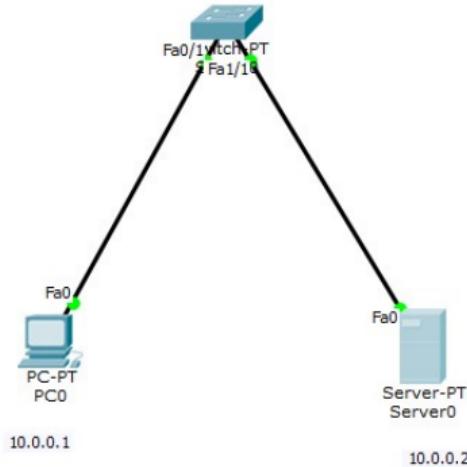
<http://csebmse/> resume.html

NAME	Lokesh
DOB	18/3/2002
COLLEGE	BMSCE

~~(Date)
9-12-2022~~







AIM :- Write a program for error detecting using CRC

```
#include <stdio.h>

void main()
{
    char input[100], key[30], temp[30], quot[100],
        rem[30], key1[30];
    printf("Enter data");
    gets(input);
    printf("Enter g(x)");
    gets(key);
    keylen = strlen(key);
    msglen = strlen(input);
    strcpy(key1, key);
    for(i=0; i<keylen-1; i++)
    {
        input[msglen+i] = '0';
    }

    for(i=0; i<keylen; i++)
        temp[i] = input[i];
    for(i=0; i<msglen; i++)
    {
        quot[i] = temp[0];
        if(quot[i] == '0')
            for(j=0; j<keylen; j++)
                key[j] = '0';
        else
            for(j=0; j<keylen; j++)
                key[j] = key1[j];
        for(j=keylen-1; j>0; j--)
        {
            if(temp[j] != key[j])
                rem[j-1] = '0';
            else
                rem[j-1] = '1';
        }
    }
}
```

```

rem [keylen-1] = input [1 + keylen];
strcpy (temp, rem);
}
strcpy (rem, temp);
printf ("Quotient is ");
for (i=0; i<msglen; i++)
printf ("%c", quot[i]);
printf ("\nRemainder");
for (i=0; i<keylen-1; i++)
printf ("%c", rem[i]);
printf (" Modified data is ");
for (i=0; i<msglen; i++)
printf ("%c", input[i]);
for (i=0; i<keylen-1; i++)
printf ("%c", rem[i]);
getch();
}

```

Output :-

Enter data :- 1011010101

Enter g(x) :- 1010

Quotient is :- 1001000100

Remainder :- 000

Modified :- 1011010101000

Checking for error

Enter data :- 1011010101000

g(x) :- 1010

Remainder :- 000

W.W.
G.T. W.W.

```
Enter data to be transmitted: 10001000000100001
```

```
Enter the Generating polynomial: 1011101
```

```
Data padded with n-1 zeros : 10001000000100001000000
```

```
CRC or Check value is : 010011
```

```
Final data to be sent : 10001000000100001010011
```

```
Enter the received data: 10001000000100001010011
```

```
Data received: 10001000000100001010011
```

```
No error detected
```

```
...Program finished with exit code 0
```

```
Press ENTER to exit console. █
```

Write a program to implement distance vector algorithm

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

int bellmanFord (int G[20][20], int v, int edge[20][20])
{ int i, u, v, k, distance[20], parent[20], s, flag = 1;
for (i = 0; i < v; i++)
    distance[i] = 1000; parent[i] = -1;
    printf("Enter source ");
    scanf("%d", &s);
    distance[s-1] = 0;
for (i = 0; i < v-1; i++)
{ for (k = 0; k < E; k++)
{ u = edge[k][0]; v = edge[k][1];
if (distance[u] + G[u][v] < distance[v])
    distance[v] = distance[u] + G[u][v];
    parent[v] = u;
}
}
for (k = 0; k < E; k++)
{ u = edge[k][0], v = edge[k][1];
if (distance[u] + G[u][v] < distance[v])
    flag = 0;
}
if (flag)
{ for (i = 0; i < v; i++)
    printf("vertex %d \rightarrow cost = %.d parent = %.d\n", i+1, distance[i], parent[i]+1);
}
return flag;
}

int main()
{ int v, edge[20][2], G[20][20], i, j, k = 0;
```

```

printf("Enter no. of vertices");
scanf("%d", &v);
printf("Enter graph in matrix form");
for(i=0; i< v; i++)
for(j=0; j< v; j++)
{
    scanf("%d", &g[i][j]);
    if(g[i][j] != 0)
        edge[0][0] = i;
        edge[0][1] = j;
}
if(Bellman-Ford(g, v, k, edge))
    printf("In no negative weight cycle");
else
    printf("In Negative weight cycle");
return 0;
}

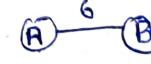
```

Output :-

Enter no. of vertices = 5

Enter graph in matrix form

0	6	0	7	0
0	0	5	8	-4
0	-2	0	0	0
0	0	-3	0	9
2	0	7	0	0



Enter source : 1

Vertex 1 → cost = 0 parent = 0

Vertex 2 → cost = 2 parent = 3

Vertex 3 → cost = 4 parent = 4

Vertex 4 → cost = 7 parent = 1

Vertex 5 → cost = -2 parent = 2

No negative weight cycle.

```
*-----*
Enter the number the routers(<10): 5
Enter 1 if the corresponding router is adjacent to routerA else enter 99:
B C D E
Enter matrix:1 1 99 99

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

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

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

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

Router Table entries for router 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 router 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 router C:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 1 99 0 1 1

Router Table entries for router 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 router E:-
Destination Router: A B C D E
Outgoing Line: A B C D E
Hop Count: 99 99 1 99 0
```

Write a program to implement dijkstra's algorithm.

```
#include <stdio.h>
#include <conio.h>
#define INFINITY 9999
#define MAX 10
void dijkstra ( int G[MAX][MAX], int n, int startnode )
int main()
{ int G[MAX][MAX], i, j, m, u;
printf ("Enter no of vertices.");
scanf ("%d", &m);
printf ("Enter adjacency matrix (%d x %d)");
for (i=0; i<n; i++)
{ for (j=0; j<n; j++)
{ scanf ("%d", &G[i][j]); }
}
printf ("Enter starting node");
scanf ("%d", &u);
dijkstra (G, m, u);
return 0;
}
```

```
void dijkstra ( int G[MAX][MAX], int m, int startnode,
{ int cost[MAX][MAX], distance[MAX], pred[MAX];
int visited[MAX], count, mindistance, nextnode,
i, j);
for (i=0; i<m; i++)
for (j=0; j<m; j++)
if (G[i][j] == 0)
cost[i][j] = INFINITY;
else
cost[i][j] = G[i][j];
for (i=0; i<m; i++)
{ distance [i] = cost [startnode][i];
}
```

```

pred[.] = startnode;
visited[.] = 0;
}

distance [startnode] = 0;
visited [startnode] = 1;
count = 1;
while (count < m-1)
{
    mindistance = INFINITY;
    for (i=0; i<n; i++)
    {
        if (distance[i] < mindistance && !visited[i])
        {
            mindistance = distance[i];
            nextnode = i;
        }
    }
    visited [nextnode] = 1;
    for (i=0; i<n; i++)
    {
        if (!visited[i])
        {
            if (mindistance[nextnode][i] < distance[i])
            {
                distance[i] = mindistance + cost[nextnode][i];
                pred[i] = nextnode;
            }
        }
    }
    count++;
    for (i=0; i<n; i++)
    {
        if (i != startnode)
        {
            printf ("\n Distance of node %d = %d ", i);
            printf (" in path = %d ", i);
            j = i;
            do
            {
                j = pred[j];
                printf (" , %d ", j);
            } while (j != startnode);
        }
    }
}

```

Output:-

Enter no. of vertices : 4

Enter adjacency matrix

	0	1	2	3
0	0	1	1	1
1	1	0	1	0
2	1	1	0	1
3	1	0	1	0

Enter starting node :: 1

Distance of 0 = 1

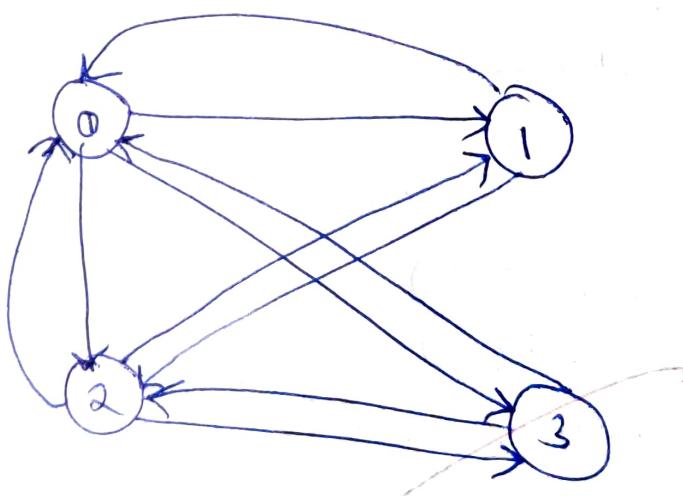
Path = 0 \leftarrow 1

Distance of 2 = 1

Path = 2 \leftarrow 1

Distance of 3 = 2

Path = 3 \leftarrow 0 \leftarrow 1



~~WAP~~
12+1/24

Enter the graph

0 9 2 5
9 0 6 8
2 6 0 0
5 8 0 0

Vertex Distance from Source
0 0
1 8
2 2
3 5

Leaky Bucket

```

#include <stdio.h>
#include <stdlib.h>
int main()
{
    int input = 0;
    int i = 0, bs;
    printf("Enter bucket limit\n");
    scanf("%d", &bs);
    int op = 1;
    printf("Bucket limit is %d", bs);
    printf("Rate is 50 mbps\n");
    while (op)
    {
        printf("Enter the input\n");
        scanf("%d", &i);
        if (i <= bs && input <= bs)
        {
            input = input + i;
            input = input - 50;
            if (input <= bs)
            {
                if (input < 0)
                    input = 0;
                printf("qty in bucket %d", input);
            }
        }
        else if (input > bs)
        {
            printf("Bucket limit exceeded");
        }
        else
        {
            printf("Bucket limit exceeded");
        }
        printf("press 1 to add input, 0 to end");
        scanf("%d", &op);
    }
    return 0;
}

```

3

Output :-

Bucket limit is 400

Rate is 50 mbps

enter input

300

qty in bucket 250

press 1 to add input 0 to end]

300

Bucket list exceeded.
press 1 to add

enter input

50

qty in bucket 25.

~~Mar
12/1/2023~~

```
Enter no of queries:3
Enter the bucket size:100
Enter output packet size:50
Enter input packet size:100
Buffer size=100 out of bucket size=100
after output new buffer size=50
Enter input packet size:150
Packet loss = 150
Buffer size=50 out of bucket size=100
after output new buffer size=0
Enter input packet size:20
Buffer size=20 out of bucket size=100
after output new buffer size=0

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

Socket Programming :-

Using TCP/IP sockets, write a client server program to make client sending file name and server to send back the contents.

- * A server has bind() method which binds specific IP and port so that it can listen to incoming requests
- * Passing an empty string means that the server can listen to incoming connections from other computers as well.
- * Server is now in listening mode.
- * At last, we make a while loop and start to accept all incoming connections and close those connections.

Client TCP.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 = clientSocket.recv(1024).decode()
print('File From Server')
print(filecontents)
clientSocket.close()
```

Output is given later, Server TCP.py

→ The server code is :-

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
```

serverSocket.listen(1)

```
while True:  
    print("Server is ready to receive")  
    connectionSocket, addr = serverSocket.accept()  
    file = open(sentence, "r")  
    l = file.read(1024)  
    connectionSocket.send(l.encode())  
    print("Sent contents' "+sentence)  
    file.close()
```

Output:-

The server is ready to receive
contents of serverTCP.py

The server is ready to receive

Enter file name :- ServerTCP.py

From Server:

```
from socket import *  
serverName = '127.0.0.1'  
serverPort = 12000  
connectionSocket.send(l.encode())  
file.close()  
connectionSocket.close()
```

```
File Edit Format Run Options Window Help
from socket import *
serverName = '127.0.0.1'
serverPort = 12000

2
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()

File Edit Shell Debug Options Window Help
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.

>>>
=====
RESTART: C:/Users/mdsur/Desktop/client.py =====

Enter file name: server.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()

>>>
```

```
File Edit Format Run Options Window Help
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()
```

```
File Edit Shell Debug Options Window Help
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022, 20:01:21) [MSC v.1934 64 bit (AMD64)]
on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=====
RESTART: C:/Users/mdsur/Desktop/server.py =====
The server is ready to receive

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

Using UDP sockets, write client server program to make client sending name and server to send back the contents

client UDP.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('In reply from server')
print(filecontents.decode("utf-8"))
clientSocket.close()
clientSocket.close()
```

Server UDP.py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print("Server ready")
while True:
    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('In sent contents of', end=' ')
    print(sentence)
    file.close()
```

Output:-

The server is ready to receive
Send contents of server.py

Enter filename : Server UDP.py

Reply from Server:

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', 12000))
serverSocket.sendto(b'bytes(1)', "utf-8", client
Address)
file.close()
```

```
C:\Users\mdsur\Desktop\UDP>python -u serverUDP.py  
The server is ready to receive  
  
Sent contents of  serverUDP.py
```

```
C:\Users\mdsur\Desktop\UDP>python -u clientUDP.py  
Enter file name: serverUDP.py  
  
Reply from Server:  
  
from socket import *  
serverPort = 12000  
serverSocket = socket(AF_INET, SOCK_DGRAM)  
serverSocket.bind(("127.0.0.1", serverPort))  
print ("The server is ready to receive")  
while 1:  
    sentence, clientAddress = serverSocket.recvfrom(2048)  
    sentence = sentence.decode("utf-8")  
    file=open(sentence,"r")  
    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()
```