VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT on

COMPUTER NETWORKS

Submitted by

VUNDAVALLI SUMA(1BM20CS192)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING

BENGALURU-560019

October-2022 to Feb-2023
(Autonomous Institution under VTU)

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 **VUNDAVALLI SUMA(1BM20CS192),** 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. Nandhini Vineeth
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

`

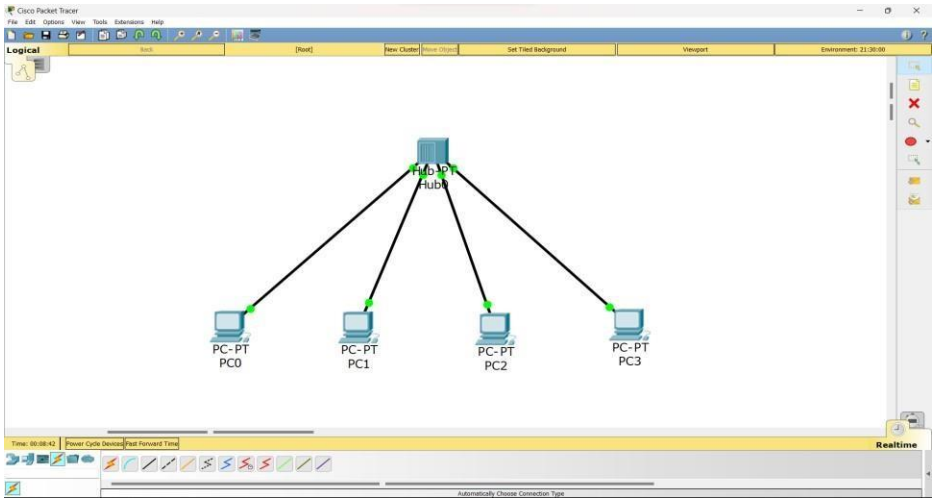
Index

Cycle-1

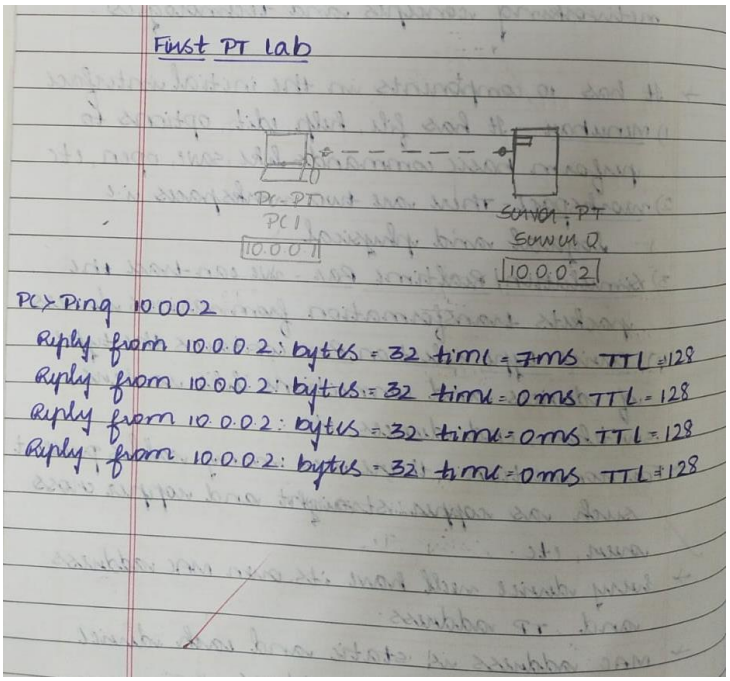
Experiment No 1

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

Hub Topology



Procedure



Date: 7/11/22

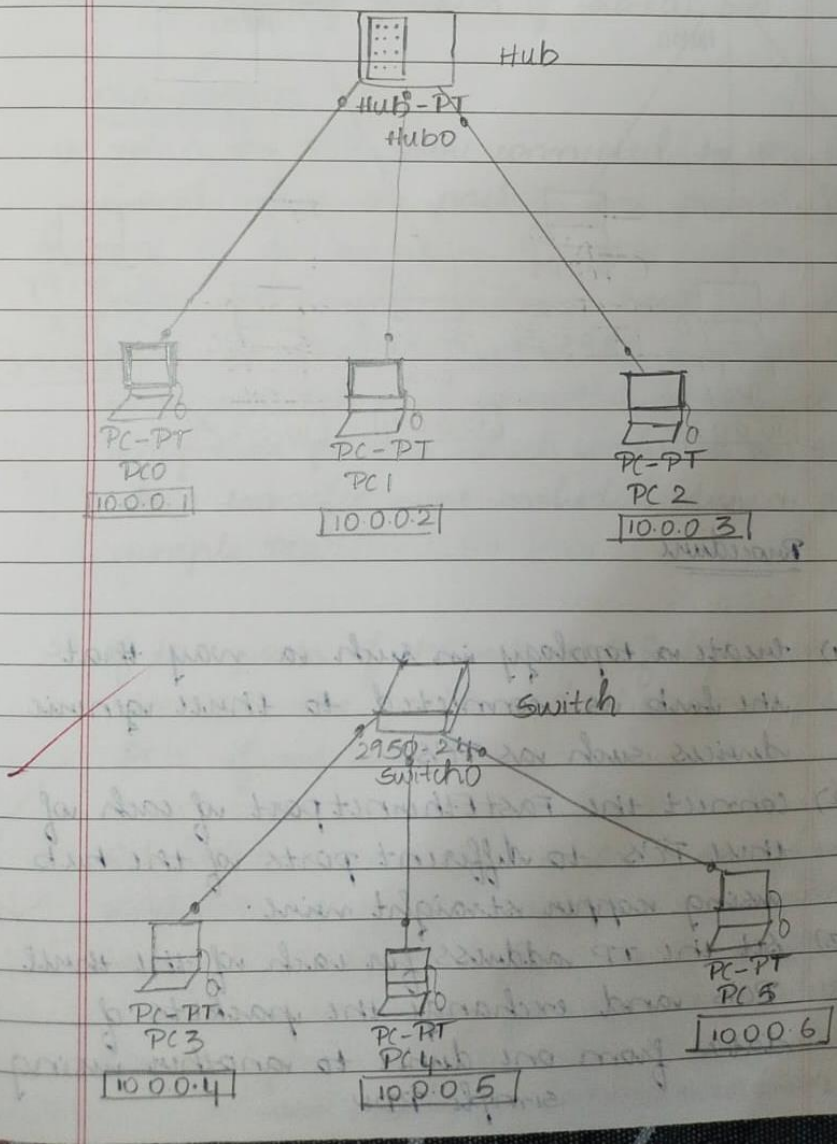
classmate
Date
Page

Experiment 1

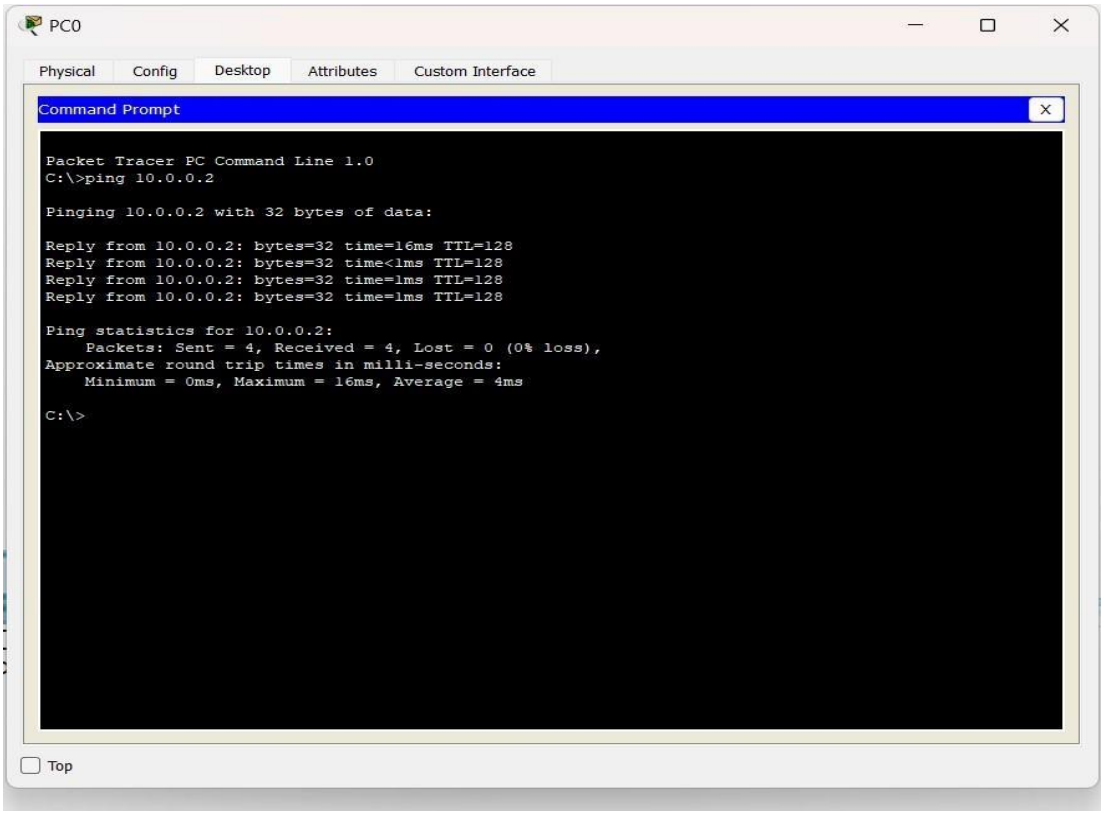
Experiment on Hubs and Switches

Aim: To create a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.

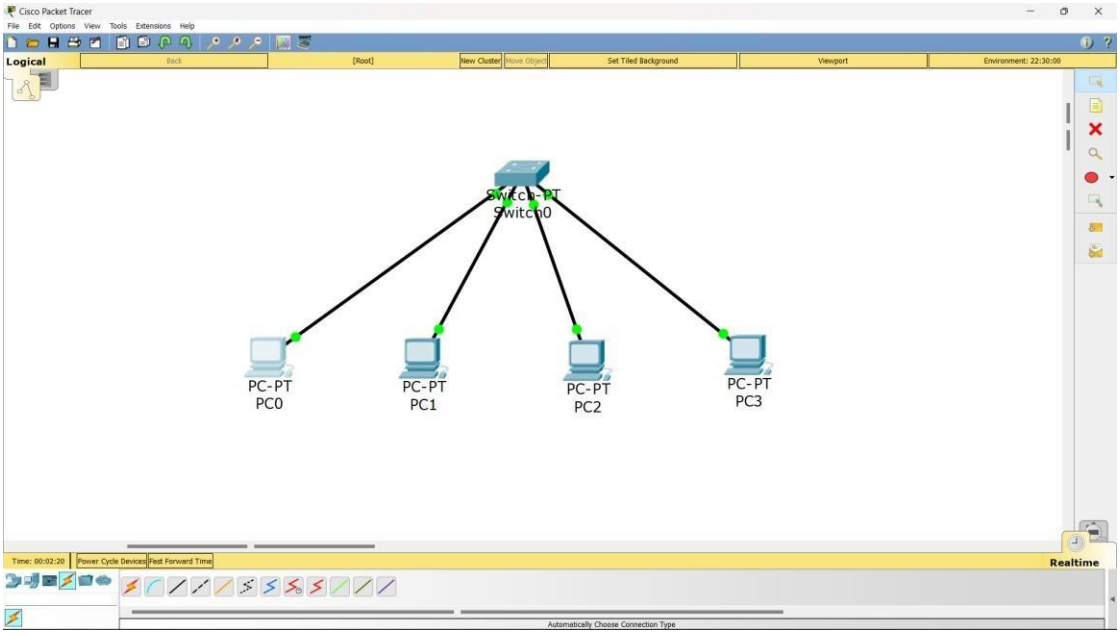
Topology



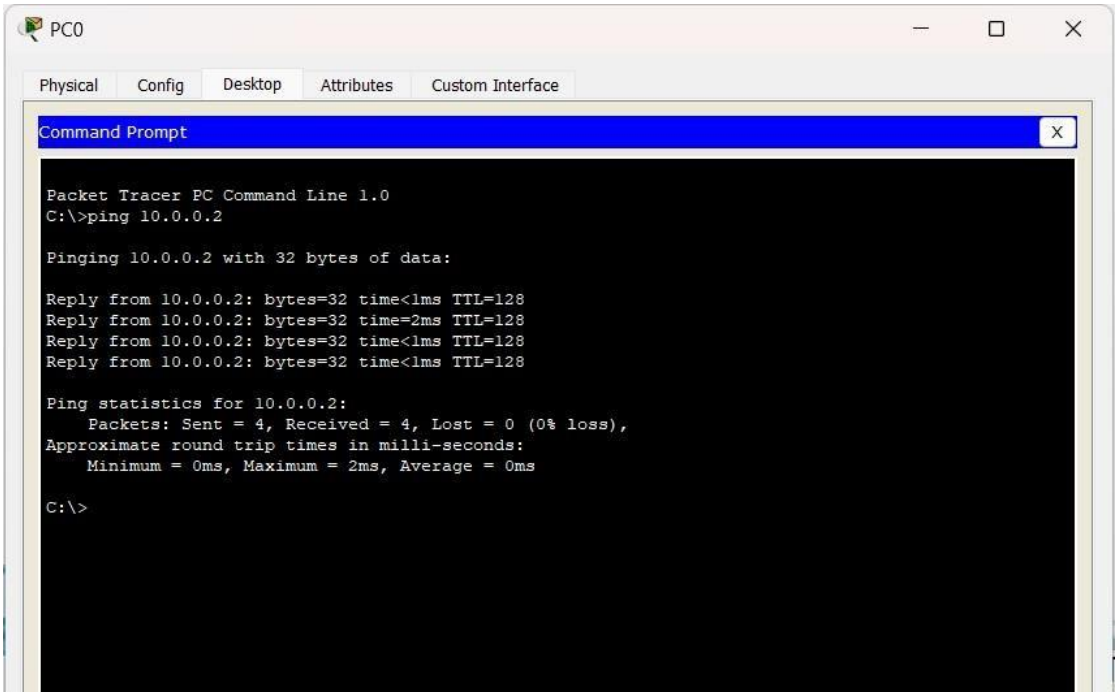
Output



Switch Topology



Output

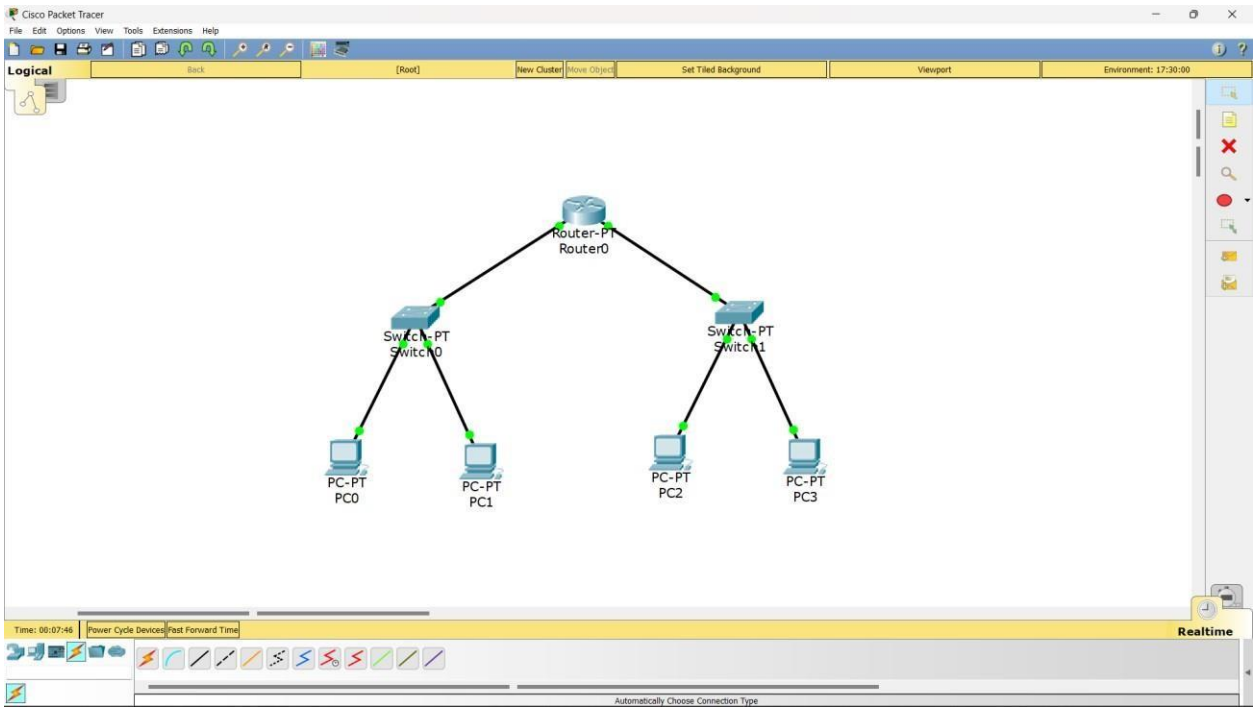

PC0
Physical
Config
Desktop
Attributes
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2
Pinging 10.0.0.2 with 32 bytes of data:
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
Reply from 10.0.0.2: bytes=32 time=2ms TTL=128
Reply from 10.0.0.2: bytes=32 time<1ms TTL=128
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

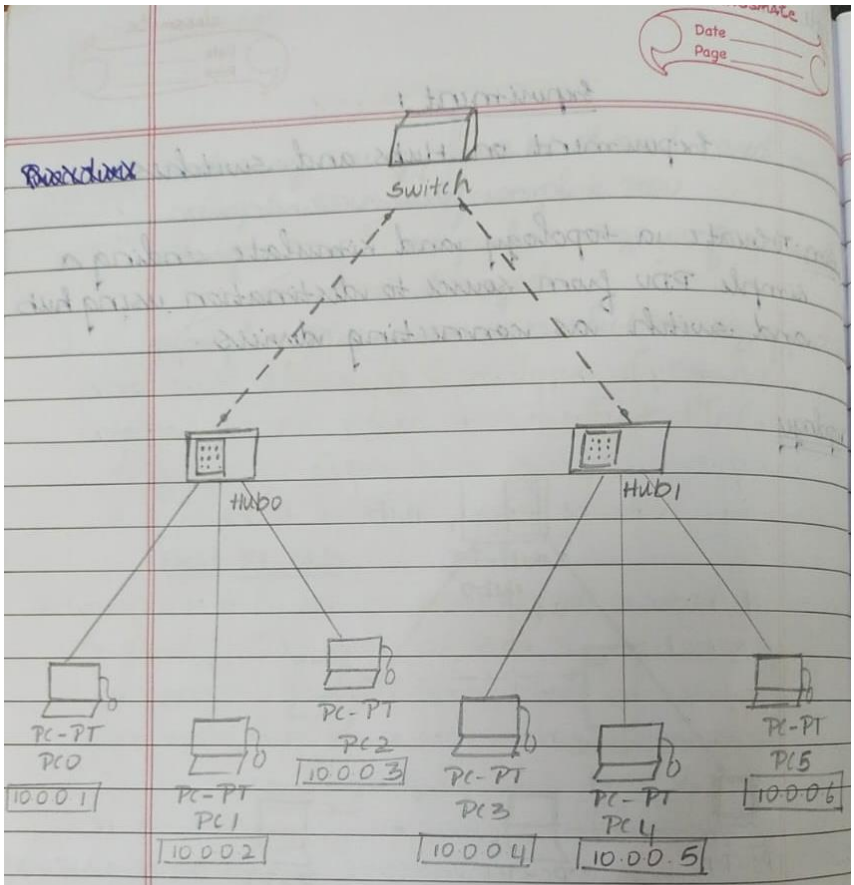
Configuring IP address to Routers in Packet Tracer. Exploring the following messages: Ping Responses, Destination unreachable, Request timed out, Reply.

Topology



Procedure

```
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)#exit
Router(config)#
Router(config)#interface FastEthernet0/0
Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#ip address 20.0.0.10 255.0.0.0
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet1/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet1/0, changed state to up

Router(config-if)#
Router(config-if)#exit
Router(config)#interface FastEthernet1/0
Router(config-if)#
```



Procedure

1) Create a topology in such a way that the hub is connected to three generic devices such as PC's.
2) Connect the FastEthernet port of each of these PC's to different ports of the hub using copper straight wire.
3) Set the IP address for each of the three PC's and exchange the packets of data from one device to another using simple PDU.

4) Repeat the same process with a switch in place of a hub and note down observations
5) then, we create a topology connecting a switch to hubs on either side using copper cross wire and then connect the PC's to the hub, set the IP address for each of them and using simple PDU, check the exchange of packets of data.

Observation

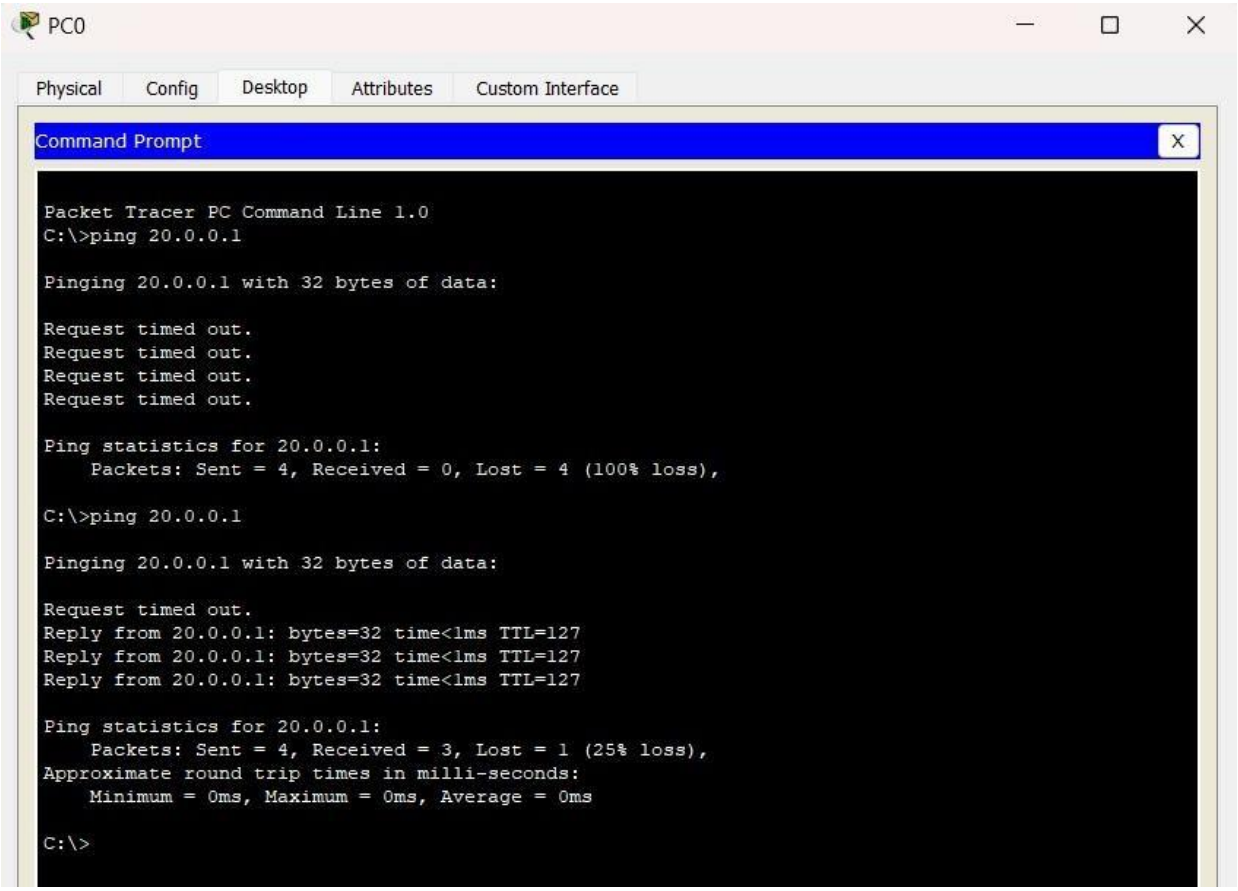
1) when the PC's were connected to the hub and when the packets are passed from one PC to another PC, the packets are also exchanged within those PC's which were not included in data transfer. But when the PC's were connected to a switch only those packets which included in the data transfer were included during the simple PDU.

2) If the ports of the hub are filled, we can increase the ports and then connect extra PC's if we want.

3) Red light during connection of the devices indicates that the connection is wrong.
Orange light / amber light - checks if the device is ready for communication
Green light - indicates that the two devices are ready for communication.

14/11/22

Output

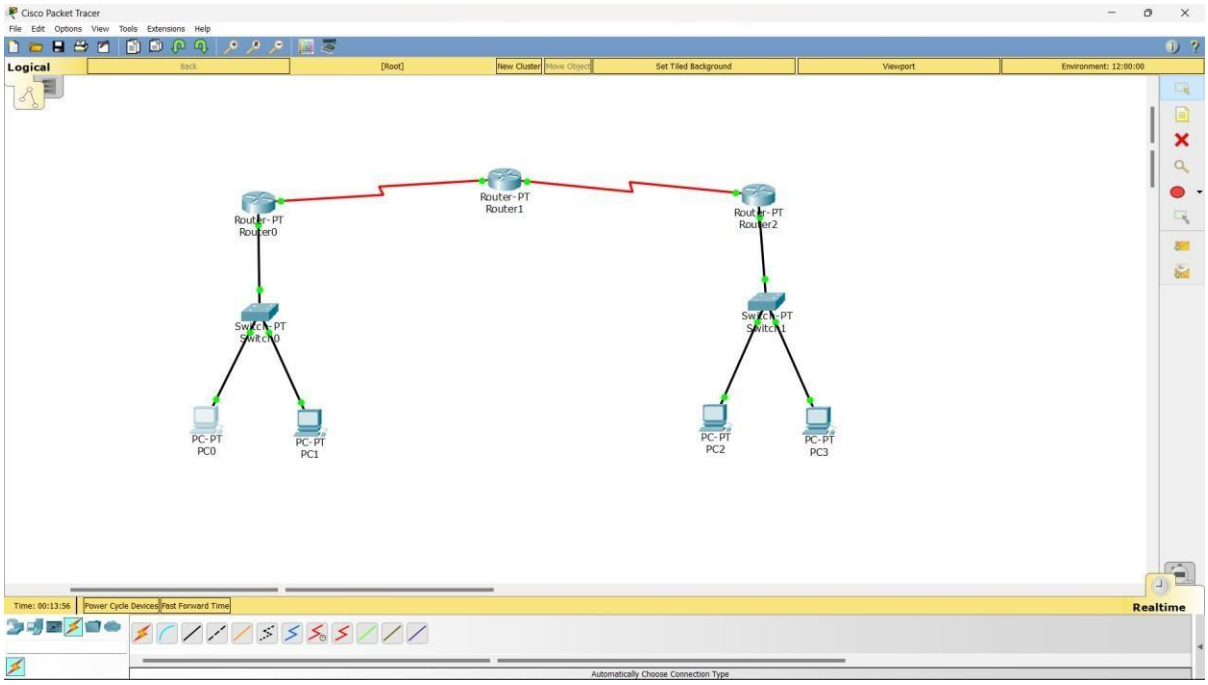

PC0
Physical
Config
Desktop
Attributes
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
C:\>ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 20.0.0.1:
Packets: Sent = 4, Received = 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

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

Gateway of last resort is not set

C    10.0.0.0/8 is directly connected, FastEthernet0/0
C    20.0.0.0/8 is directly connected, Serial2/0

Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip route 0.0.0.0 0.0.0.0 20.0.0.2
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console

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

Gateway of last resort is 20.0.0.2 to network 0.0.0.0

C    10.0.0.0/8 is directly connected, FastEthernet0/0
C    20.0.0.0/8 is directly connected, Serial2/0
S*   0.0.0.0/0 [1/0] via 20.0.0.2
```

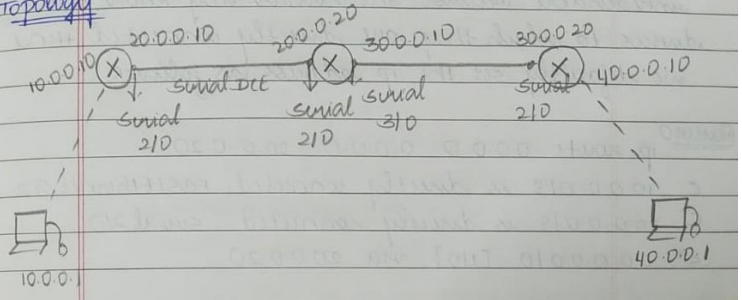
Date: 19/11/22

classmate
Date
Page

Experiment 3

Aim :- To configure default route to the router.

Topology

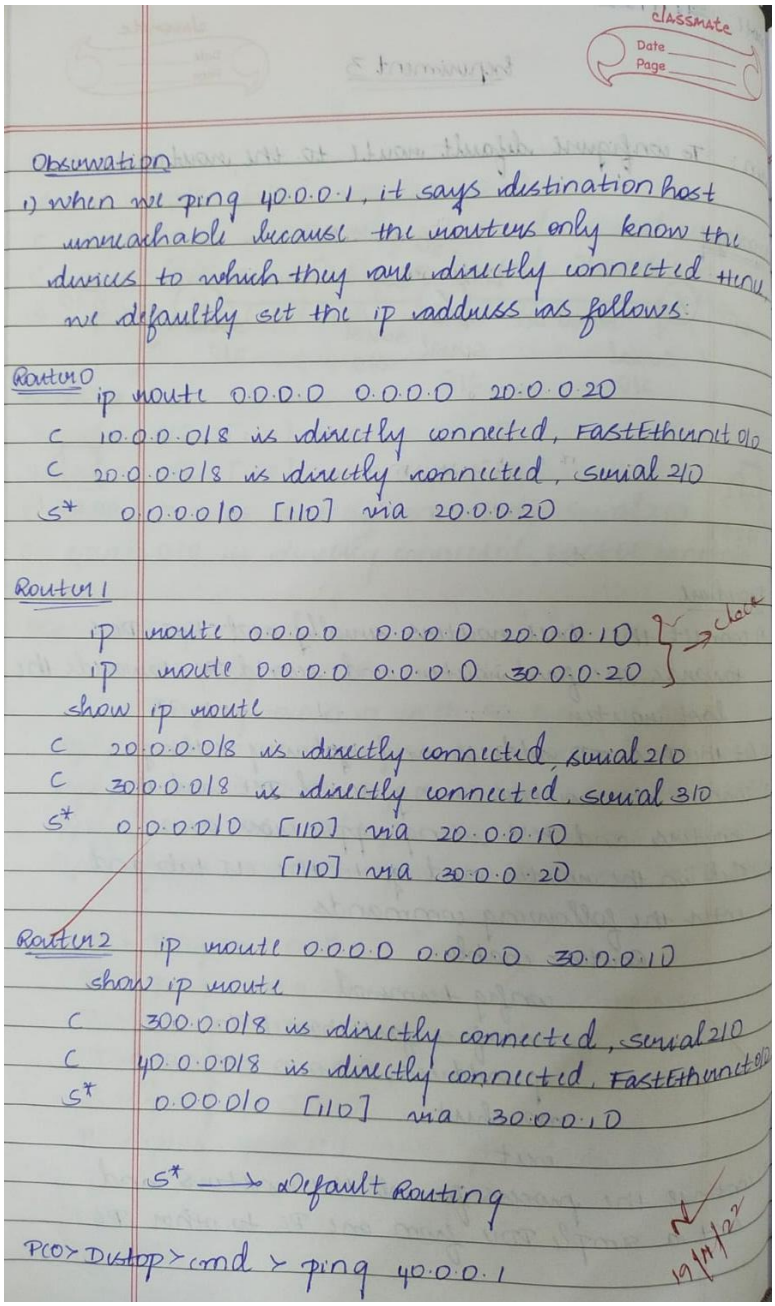


Procedure

1) Connect the three routers serially and two PC's towards the first router and second PC towards the last router.
2) Set the PC's IP address and gateway settings
3) Connect the routers using serial DCE and the routers and PC's using copper cross-over.
4) Click on the router and go to the CLI tab and enter the following commands

```
Router> enable
        config terminal
        interface fastEthernet 0/0
        ip address 10.0.0.10 255.0.0.0
        no shut
        exit
```

5) Continue the process for all the routers and sent a simple PDU from one PC to other PC.



Observation

1) when we ping 40.0.0.1, it says destination host unreachable because the routers only know the devices to which they are directly connected that we defaultly set the ip addresses as follows:

Router0

ip route 0.0.0.0 0.0.0.0 20.0.0.20

C 10.0.0.0/8 is directly connected, FastEthernet 0/0

C 20.0.0.0/8 is directly connected, Serial 2/0

S* 0.0.0.0/0 [1/0] via 20.0.0.20

Router1

ip route 0.0.0.0 0.0.0.0 20.0.0.10 } → clock

ip route 0.0.0.0 0.0.0.0 30.0.0.20 }

show ip route

C 20.0.0.0/8 is directly connected, Serial 2/0

C 30.0.0.0/8 is directly connected, Serial 3/0

S* 0.0.0.0/0 [1/0] via 20.0.0.10

[1/0] via 30.0.0.20

Router2 ip route 0.0.0.0 0.0.0.0 30.0.0.10

show ip route

C 300.0.0/8 is directly connected, Serial 2/0

C 40.0.0.0/8 is directly connected, FastEthernet 0/0

S* 0.0.0.0/0 [1/0] via 30.0.0.10

S* → Default Routing

PC0> Desktop > cmd > ping 40.0.0.1

19/11/21

Output

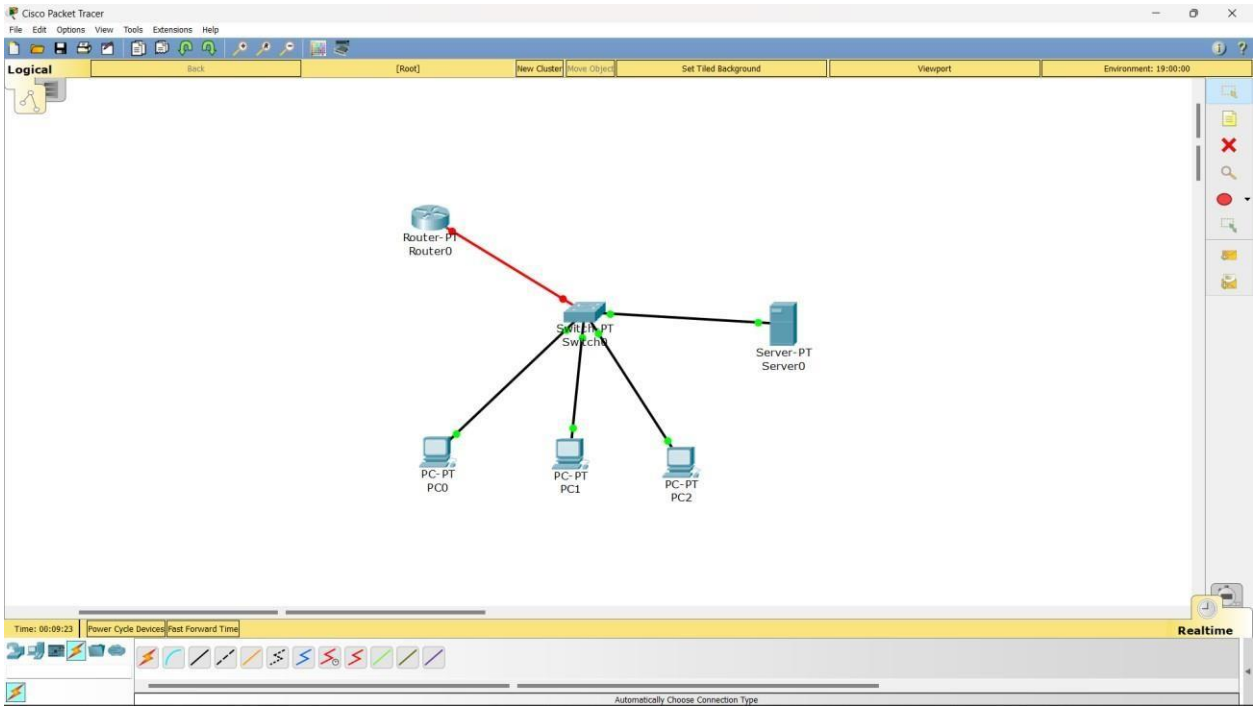
```
Packet Tracer PC Command Line 1.0
C:\>ping 40.0.0.1

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

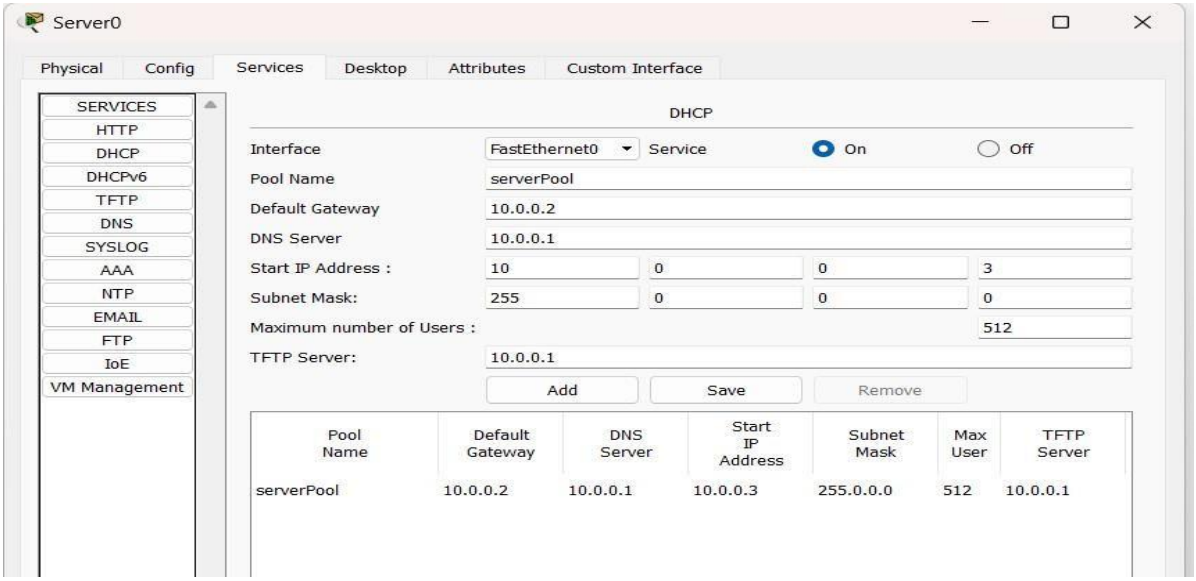
Aim of the program

Configuring DHCP within a LAN in a packet Tracer

Topology



Procedure



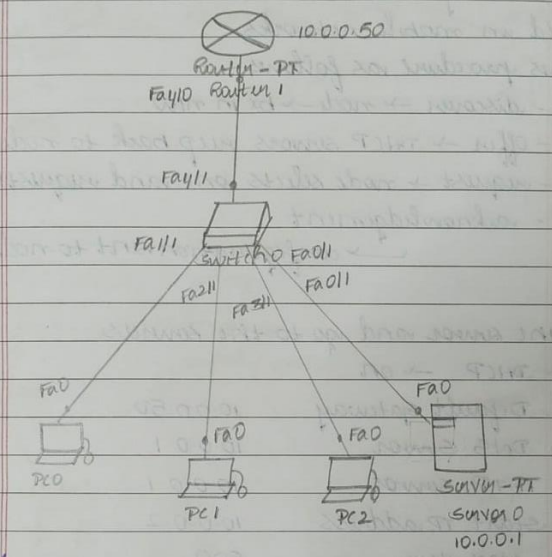
Date: 28/11/22

classmate Date Page

Experiment 4

Aim: Configuring DHCP within a LAN in a packet Tracer.

Topology



Procedure

1) Connect the devices according to above topology.
2) Set the IP address of the server as 10.0.0.1 and gateway settings to 10.0.0.50.
3) Configure the router using the following commands.

```
Router> CLI tab
        enable
        config terminal
        Intufac fastethernet4/0
        ip address 10.0.0.50 255.0.0.0
        no shut
        exit
```

Dynamic Host configuration Protocol (DHCP)

- It dynamically allocates IP address
- applied in mobile networks
- follows procedure as follows
  - D - discover → node → bx in n/w
  - O - Offer → DHCP servers reply back to node
  - R - request → node selects one and requests
  - A - acknowledgement
    → configuration sent to node.

1) Click on the server and go to the services
   → DHCP → on

Default Gateway	10.0.0.50
DNS Server	10.0.0.1
TFTP Server	10.0.0.1
Start IP address	10.0.0.2
Max. User	500

→ click on [save]

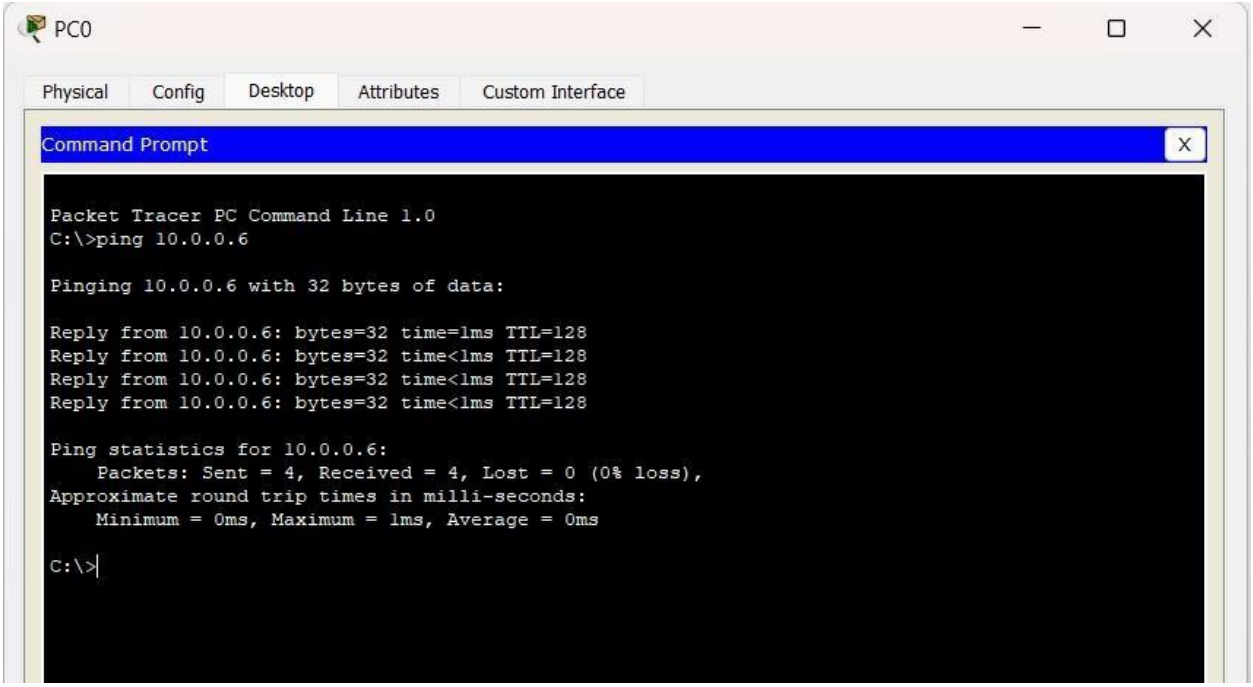
PC0 → Desktop > IP config > DHCP > DHCP request successful
PC1 → Desktop > IP config > DHCP > DHCP request successful
PC2 → Desktop > IP config > DHCP > DHCP request successful

Observations

Ping 10.0.0.4

Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 120
Reply from 10.0.0.4: bytes = 32 time = 3ms TTL = 120
Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 120
Reply from 10.0.0.4: bytes = 32 time = 0ms TTL = 120

Output

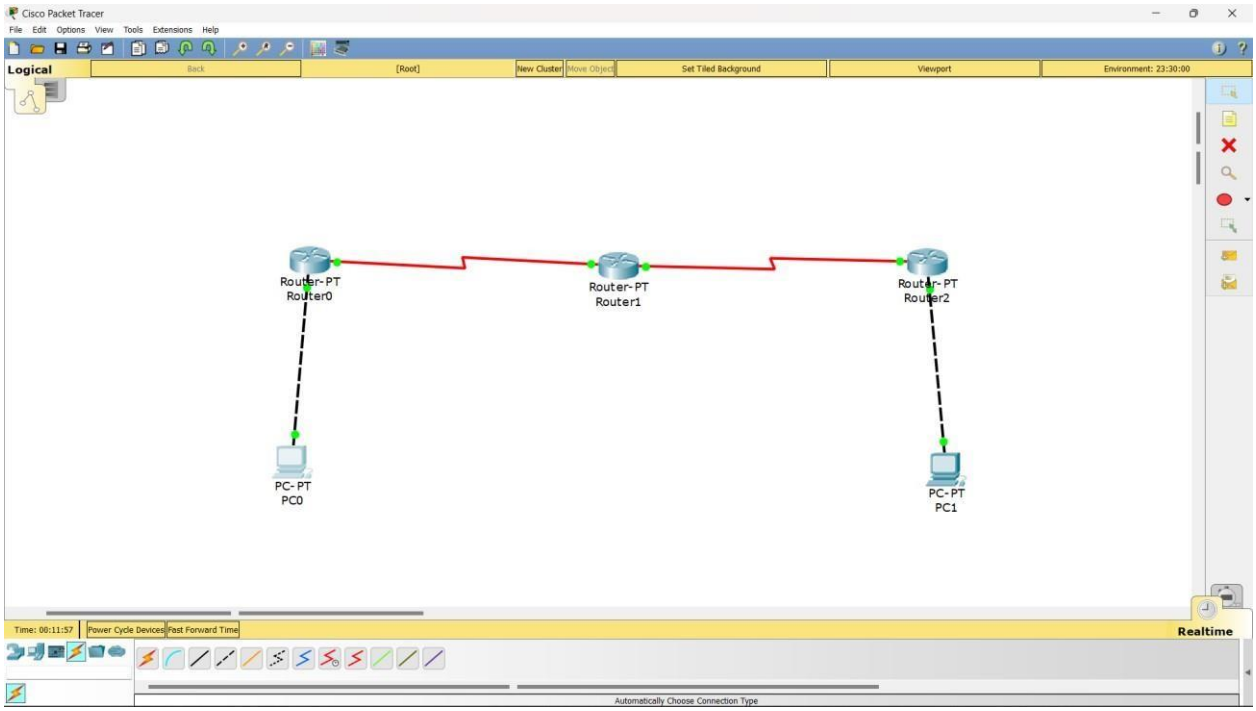

PC0
Physical
Config
Desktop
Attributes
Custom Interface
Command Prompt
X
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.6
Pinging 10.0.0.6 with 32 bytes of data:
Reply from 10.0.0.6: bytes=32 time=1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
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
```

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:\>
```

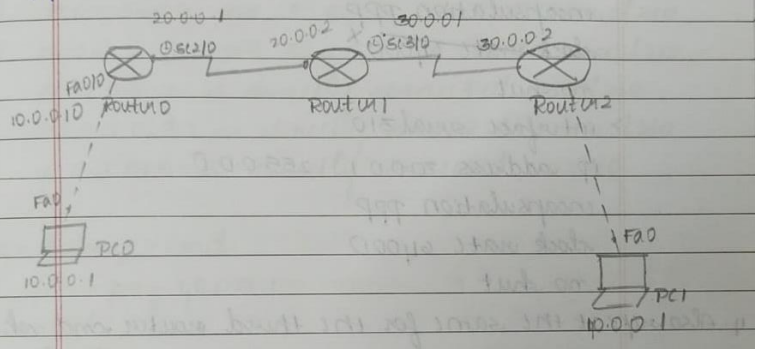
Date: 5/12/2022

classmate Date Page

Routing Information Protocol (RIP)

Aim: To configure RIP Routing Protocol in Routers.

Topology:



Procedure

1. connect the routers and PC's according to the above topology. connect the two routers with a clocked serial DCE connection.
2. Set the IP address and the gateway for both the PC's.
3. Router0 > enable
   config t
   interface fastethernet 0/0 ~~10.0.~~
   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 (point-to-point protocol)
   clock rate 64000
   no shut

Router1> enable
config t
intfaci serial 2/0 ~~0/0/2~~
ip address 20.0.0.2 255.0.0.0
encapsulation PPP
~~clock rate 64000~~ ×
no shut
> intfaci serial 3/0
ip address 30.0.0.1 255.0.0.0
encapsulation PPP
clock rate 64000
no shut

4. Also repeat the same for the third router and when we ping we get Distination host unreachable.

Router0> router rip
~~#~~ network 10.0.0.0
~~#~~ network 20.0.0.0
exit

Router1> router rip
network 20.0.0.0
network 30.0.0.0
exit

Router2> router rip
network 30.0.0.0
network 40.0.0.0
exit

Observations

1) Two types in RIP
   *) RIP-1 (does not support CIDR)
   *) RIP-2 (supports CIDR)

2) After we run the following commands

Router2> # show ip route

R 10.0.0.0/8 via 30.0.0.1 00:00:17, Serial2/0
R 20.0.0.0/8 via 30.0.0.1, 00:00:17, Serial2/0
C 30.0.0.0/8 is directly connected, Serial2/0
C 30.0.0.0/32 is directly connected, Serial2/0
C 40.0.0.0/8 is directly connected, FA0/0

PC0> Desktop > cmd

ping 40.0.0.1

Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=2ms TTL=125
Reply from 40.0.0.1: bytes=32 time=19ms TTL=125
Reply from 40.0.0.1: bytes=32 time=16ms TTL=125

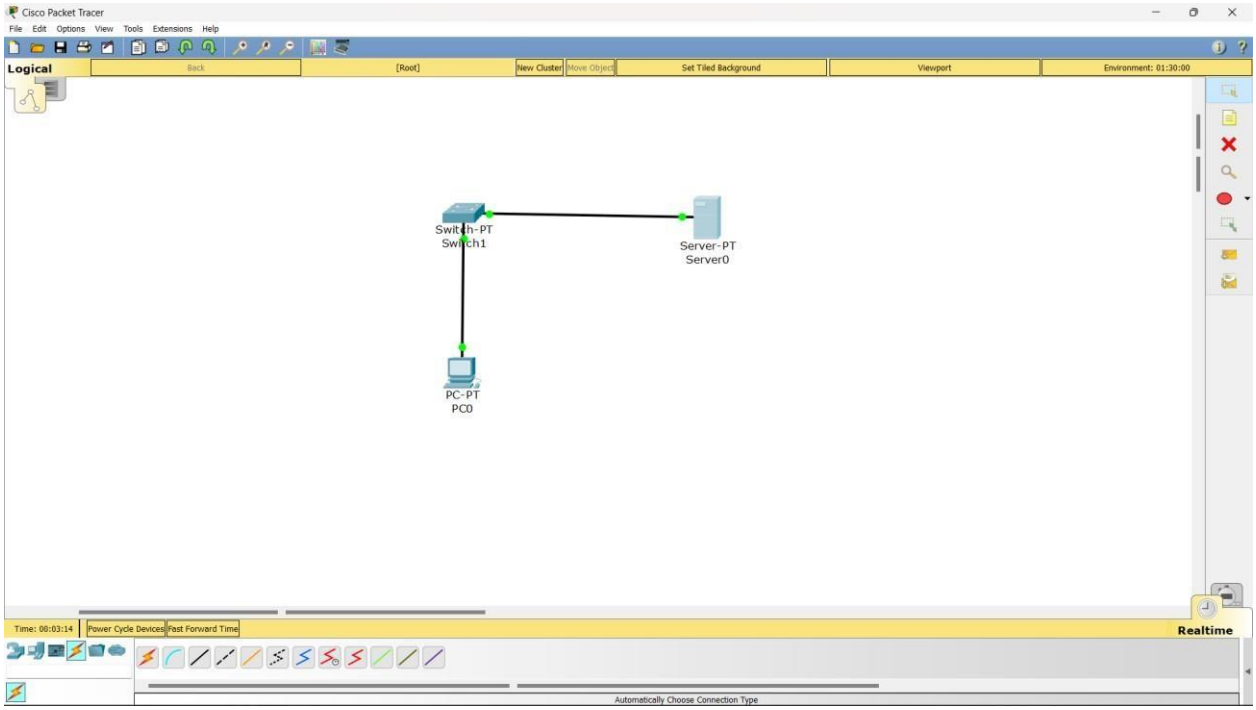
28/11/22

Experiment No 6

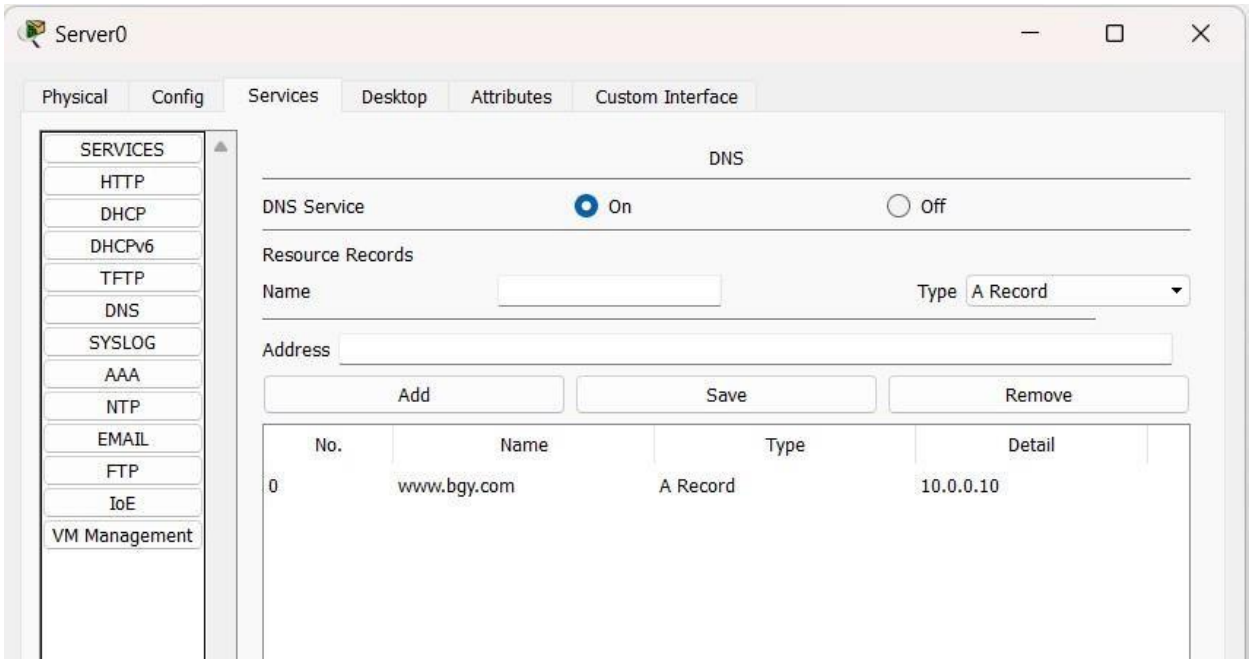
Aim of the program

Demonstration of WEB server and DNS using Packet Tracer

Topology



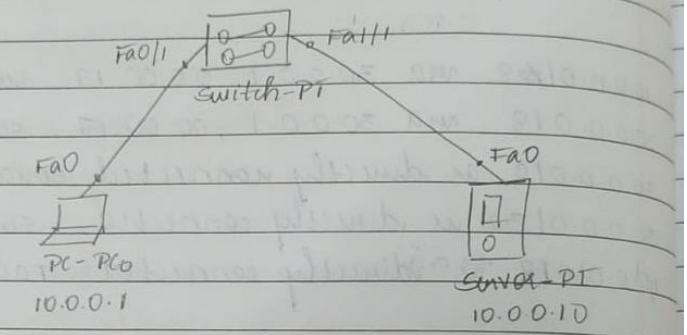
Procedure



Date: 12/12/2022

LAB 6

Aim: Demonstration of WEB Server and DNS using packet Tracer.



Procedure

1. A generic PC, generic Server-PT and a switch PT are joined as above.
2. Set the IP address for the PC as well as server.
3. In the services tab of the server, HTTP is switched on and DNS is on.
5. In DNS section of server, assign a name and IP address 10.0.0.10 and click on ADD
6. PC > Desktop > Web browser
   a) Enter IP address of server (10.0.0.10) in the URL and click enter.

Web Browser

URL: https://10.0.0.10

Welcome to Cisco Packet Tracer.

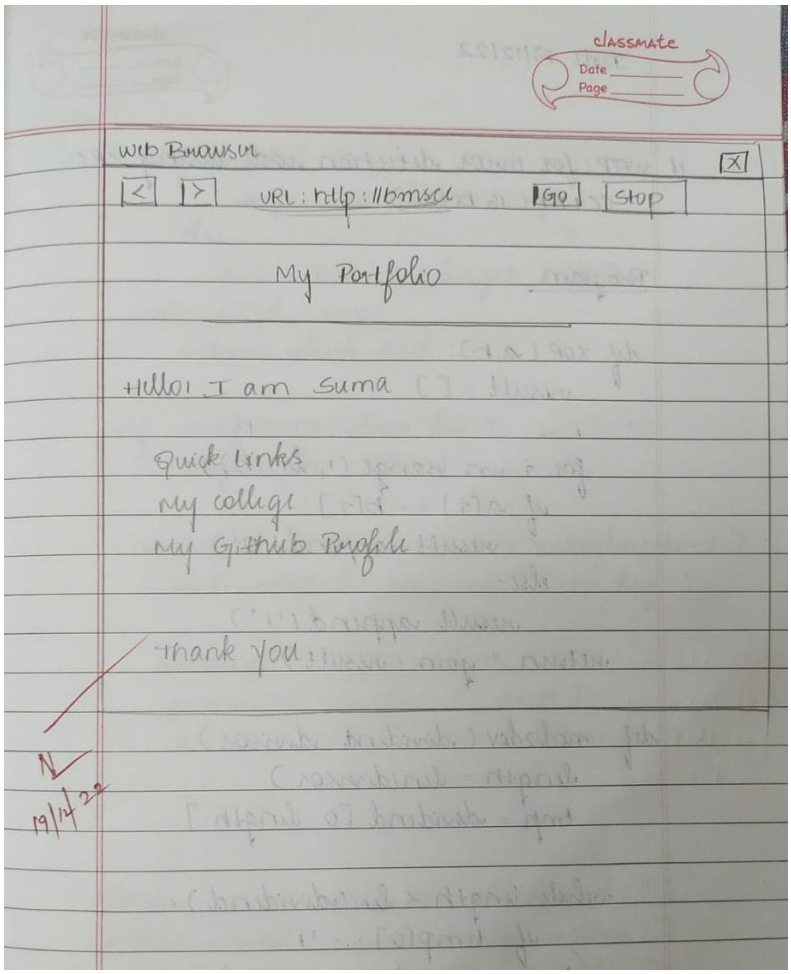
Quick Links:

A small Page

Copyrights

Image Page

Images


classmate
Date
Page
Web Browser
URL : http://bmscl
Go
Stop
My Portfolio
Hello! I am Suma
Quick Links
my collge
My Github Profile
Thank you
19/12/22

Output


PC0
Physical
Config
Desktop
Attributes
Custom Interface
Web Browser
X
URL
http://www.bgy.com/index.html
Go
Stop
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<bits/stdc++.h> using
namespace std; void receiver(string
data, string key);


string xor1(string a, string b)
{

        string result = "";


        int n = b.length();


        for(int i = 1; i < n; i++)
        {
                if (a[i] == b[i])
                        result += "0";
else
                        result += "1";
        }
```

```
        return result;
}

string mod2div(string dividend, string divisor)
{
        int pick = divisor.length();

        string tmp = dividend.substr(0, pick);

        int n = dividend.length();

        while (pick < n)
        {
                if (tmp[0] == '1')                      tmp =
xor1(divisor, tmp) + dividend[pick];
                else
                        tmp = xor1(std::string(pick, '0'), tmp) +
                                dividend[pick];

                pick += 1;
        }
        if (tmp[0] == '1')
tmp = xor1(divisor, tmp);
        else
                tmp = xor1(std::string(pick, '0'), tmp);

        return tmp;
}
```

```
void encodeData(string data, string key)
{
	int l_key = key.length();

	string appended_data = (data +std::string(l_key - 1, '0'));

	string remainder = mod2div(appended_data, key);
```

```
        string codeword = data + remainder;
cout << "Remainder : "
                << remainder << "\n";
        cout << "Encoded Data (Data + Remainder) :"
                << codeword << "\n";
        receiver(codeword, key);
}
void receiver(string data, string key)
{       string currxor = mod2div(data.substr(0, key.size()),
key);   int curr = key.size();    while (curr != data.size())
        {
                if (currxor.size() != key.size())
                {
                        currxor.push_back(data[curr++]);
                }
                else
                {
                        currxor = mod2div(currxor, key);
                }
        }

        if (currxor.size() == key.size())
        {
                currxor = mod2div(currxor, key);
        }
        if (currxor.find('1') != string::npos)
        {       cout << "there is some error in data" <<
        endl;
```

```
        }
        else
        {
                cout << "correct message recieved" << endl;
        }
} int
main()
{
```

```
string data = "1011101";string
key = "100010000001";

        encodeData(data, key);


        return 0;
}
```

Output

```
Remainder : 10001011000
Encoded Data (Data + Remainder) :101110110001011000
correct message recieved


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

Experiment No 2

Aim of the Experiment

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

Code

```
#include<stdio.h>



define INF  99999
#define n 5


void printSolution(int g[n])
{     printf("Hop count         :
");     for(int j=0;j<n;j++)
    {         if(g[j] ==
INF)
printf("INF\t");
else
            printf("%d\t",g[j]);
    }
printf("\n");
}


void findShortestPath(int dist[][n])
{


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

```
        {               if(dist[i][j] > dist[i][k] +
dist[k][j]
            &&(dist[i][k] != INF && dist[k][j] != INF))
            {                   dist[i][j] = dist[i][k]
+ dist[k][j];
            }
        }
```

```
        }
    }

    char c = 'A'; for(int
i=0; i<n; i++ )
    {    printf("Router table entries for router %c:\n",c);
printf("Destination router: A\tB\tC\tD\tE\n");
printSolution(dist[i]);  c++;
    }
}

int main() {
    int graph[][n] = { {0,  1,  1, INF, INF},
                       {1,  0,  INF,  INF, INF},
                       {1, INF, 0, 1,  1},
                       {INF, INF, 1, 0, INF},
                       {INF, INF, 1, INF, 0}};

    findShortestPath(graph);
return 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.

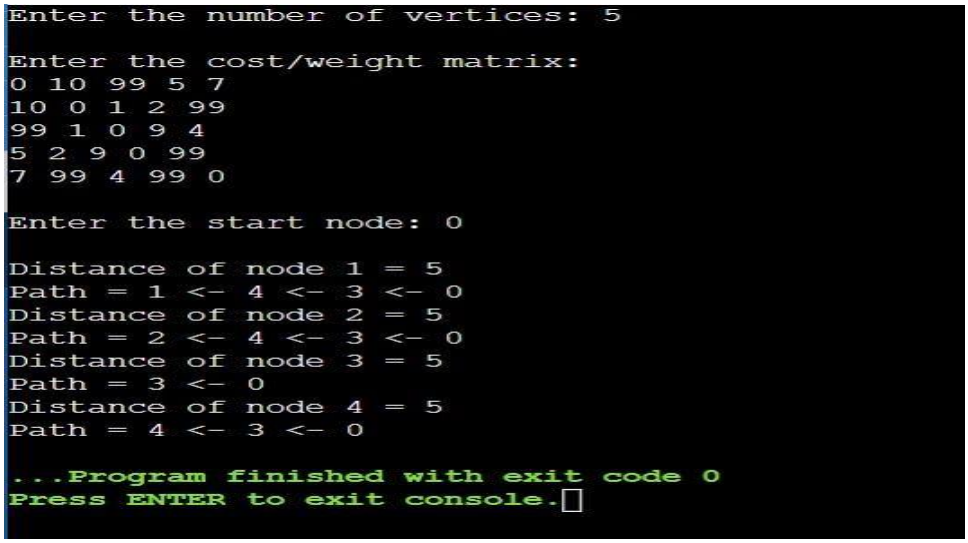
Code

```
#include <stdio.h>
#include <stdlib.h>
void dijkstra(int graph[10][10],int V)
{
   int distance[V], predefine[V], visited[V];     int
startnode, count, min_distance, nextnode, i, j;
printf("\nEnter the start node: ");     scanf("%d",
&startnode);     for(i=0; i<V; i++) {
distance[i] = graph[startnode][i];         predefine[i]
= startnode;         visited[i] = 0;
   }
   distance[startnode] = 0;
visited[startnode] = 1;     count
= 1;     while(count<V-1) {
min_distance = 99;
for(i=0; i<V; i++) {
          if(distance[i] < min_distance && visited[i]==0)
          {
             min_distance = distance[i];
             nextnode = i;
          }
       }
       visited[nextnode] = 1;
for(i=0;i<V;i++)
       {
```

```
if(visited[i] == 0)
          {
             if((min_distance + graph[nextnode][i]) < distance[i])
             {


distance[i] = min_distance + graph[nextnode][i];predefine[i]
= nextnode;
             }
          }
       }
       count = count + 1;
   }
   for(i=0;i<V;i++) {         if(i!=startnode) {
printf("\nDistance of node %d = %d", i, distance[i]);
printf("\nPath = %d",i);
          j = i;
do            {
             j = predefine[j];
printf(" <- %d",j);
          } while (j != startnode);
       }
   }
} int
main()
{
   int i, j;     int V;     printf("Enter the number
of vertices: ");     scanf("%d", &V);     int
graph[V][V];     printf("\nEnter the
cost/weight matrix: \n");     for(i=0; i<V; i++)
```

```
{       for(j=0;j<V;j++) {          scanf("%d",
&graph[i][j]);
      }

dijkstra(graph, V);
return 0;
}
```

Output:



Experiment No 4

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

Server:

```
from socket import * serverName = '' serverPort =
12530 serverSocket =
socket(AF_INET,SOCK_STREAM)
serverSocket.bind((serverName,serverPort))
serverSocket.listen(1) print("The server is ready to
receive") while 1:
   connectionSocket, addr = serverSocket.accept()
sentence = connectionSocket.recv(1024).decode()     try:
      file = open(sentence,"r")        l =
file.read(1024)
```

connectionSocket.send(l.encode())

file.close() except Exception as e:

message = "No such file exist"

connectionSocket.send(message.encode()) connectionSocket.close()

Client: from socket import *

serverName = '192.168.1.104'

serverPort = 12530

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 ('From

Server:', filecontents) clientSocket.close()

Output

```
Enter file namemain.cpp
From Server: #include <bits/stdc++.h>
using namespace std

class Node{

        bool color = 0; // 1 -> black; 0 -> red
        Node *left = NULL;
        Node *right = NULL;
        Node *parent = NULL;
        int key;

        Node(int k)
        {
                key = k;
        }

};
```

Experiment No 5

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

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)


    file=open(sentence,"r")

l=file.read(2048)
```

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

print("sent back to client",l) file.close() Client:

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()

Output

```
Enter file namemain.cpp
From Server: b'#include <bits/stdc++.h>\nusing namespace std\n\nclass Node{\n\t\n\tbool color = 0; // 1 -> black; 0 -> r
ed\n\tNode *left = NULL;\n\tNode *right = NULL;\n\tNode *parent = NULL;\n\tint key;\n\t\n\tNode(int k)\n\t{\n\t\tkey = k
;\n\t}\n\t\n};\n\n\nvoid inorderTraversal(Node *head)\n{\n\tif(head != NULL)\n\t{\n\t\tinorderTraversal(head->left);\n\t
\tcout<<head->key<< "(" << head->color << ") ";\n\t\tinorderTraversal(head->right);\n\t}\n}\n\n\nNode* leftRotate(Node *
x)\n{\n\tNode *y = x->right;\n\tx->right = y->left;\n\t\n\tif(x->right != NULL)\n\t{\n\t\tx->right->parent = x;\n\t}\n\t
\n\tif(x->parent == NULL)\n\t\ty->parent = NULL;\n\telse\n\t{\n\t\ty->parent = x->parent;\n\t\tif(x == x->parent->left)\
n\t\t\tx->parent->left = y;\n\t\telse\n\t\t\tx->parent->right = y;\n\t}\n\ty->left = x;\n\tx->parent = y;\n\t\n\treturn
y;\n}\n\nNode* rightRotate(Node *y)\n{\n\tNode *x = y->left;\n\ty->left = x->right;\n\t\n\tif(y->left != NULL)\n\t{\n\t\
ty->left->parent = y;\n\t}\n\t\n\tif(y->parent == NULL)\n\t{\n\t\tx->parent = NULL;\n\t}\n\telse\n\t{\n\t\tx->parent = y
->parent;\n\t\tif(y == y->parent->left)\n\t\t\ty->parent->left = x;\n\t\telse\n\t\t\ty->parent->right = x;\n\t}\n\ty->pa
rent = x;\n\tx->right = y;\n\t\n\treturn x;\n}\n\nNode* bstInsert(Node *head, int val)\n{\n\tNode *newNode = new Node(va
l);\n\tif(head == NULL)\n\t{\n\t\thead = newNode;\n\t}\n\telse\n\t{\n\t\tNode *curr = head;\n\t\tNode *prev = NULL;\n\t\
t\n\t\twhile(curr != NULL)\n\t\t{\n\t\t\tprev = curr;\n\t\t\tif(val < curr->key)\n\t\t\t\tcurr = curr->left;\n\t\t\telse
\n\t\t\t\tcurr = curr->right;\n\t\t}\n\t\t\n\t\tif(val < prev->key)\n\t\t\tprev->left = newNode;\n\t\telse\n\t\t\tprev->
right = newNode;\n\t}\n\t\n\treturn head;\n}\n\n\nint main ()\n{\n\t\n\tNode *head = NULL;\n\tint n;\n\tint k;\n\t\n\tco
ut<<"Enter the number of elements: ";\n\tcin>>n;\n\tcout<<"Enter the elements: ";\n\t\n\tfor(int i=0; i<n; i++)\n\t{\n\t
\tcin>>k;\n\t\thead = bstInsert(head, k);\n\t}\n\t\n\tleftRotate(head);\n\tinorderTraversal(head);\n\t\n\treturn 0;\n}'
```