

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JnanaSangama”, Belgaum -590014, Karnataka.



LAB REPORT on

COMPUTER NETWORKS

Submitted by

ANIKSHA S (1BM20CS015)

in partial fulfilment 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 **ANIKSHA S(1BM20CS015)**, who is bonafide student of **B.M. S. College of Engineering**. It is in partial fulfilment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum 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. SHYAMALA G
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	17/11/22	Creating a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.	1
2	17/11/22	Configuring IP address to Routers in Packet Tracer. Exploring the following messages: Ping Responses, Destination unreachable, Request timed out, Reply	4
3	1/12/22	Configuring static and default route to the Router	6
4	8/12/22	Configuring DHCP within a LAN in a packet Tracer	9
5	15/12/22	Configuring RIP Routing Protocol in Routers	11
6	15/12/22	Demonstration of WEB server and DNS using Packet Tracer	14
7	29/12/22	Write a program for error detecting code using CRC-CCITT (16-bits).	16
8	13/1/23	Write a program for distance vector algorithm to find suitable path for transmission.	20
9	5/1/23	Implement Dijkstra's algorithm to compute the shortest path for a given topology.	23
10	30/1/23	Write a program for congestion control using leaky bucket algorithm.	26
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.	28
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.	30

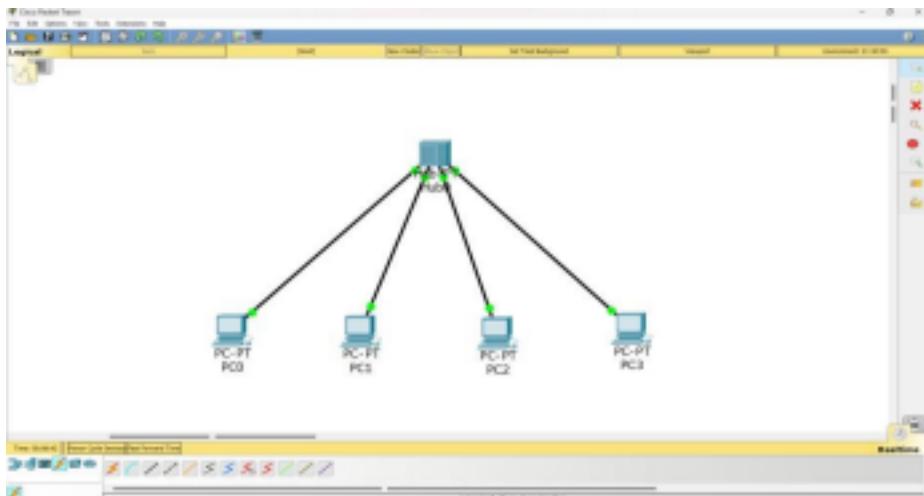
Cycle-1

Experiment No 1

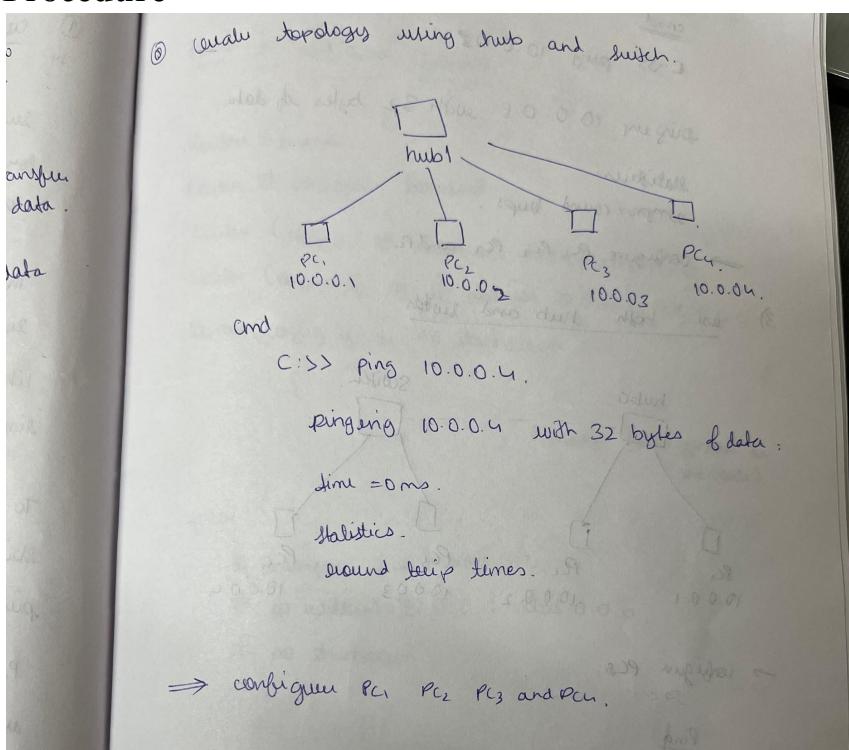
Aim of the program

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

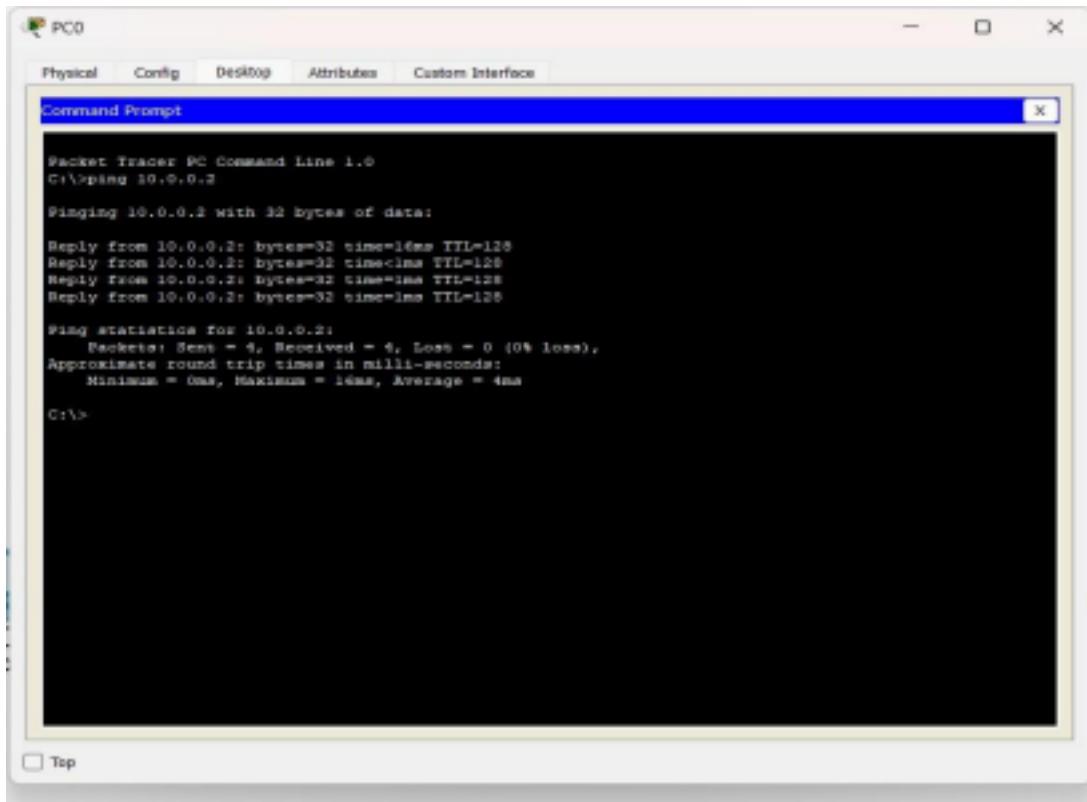
Hub Topology



Procedure



Output



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

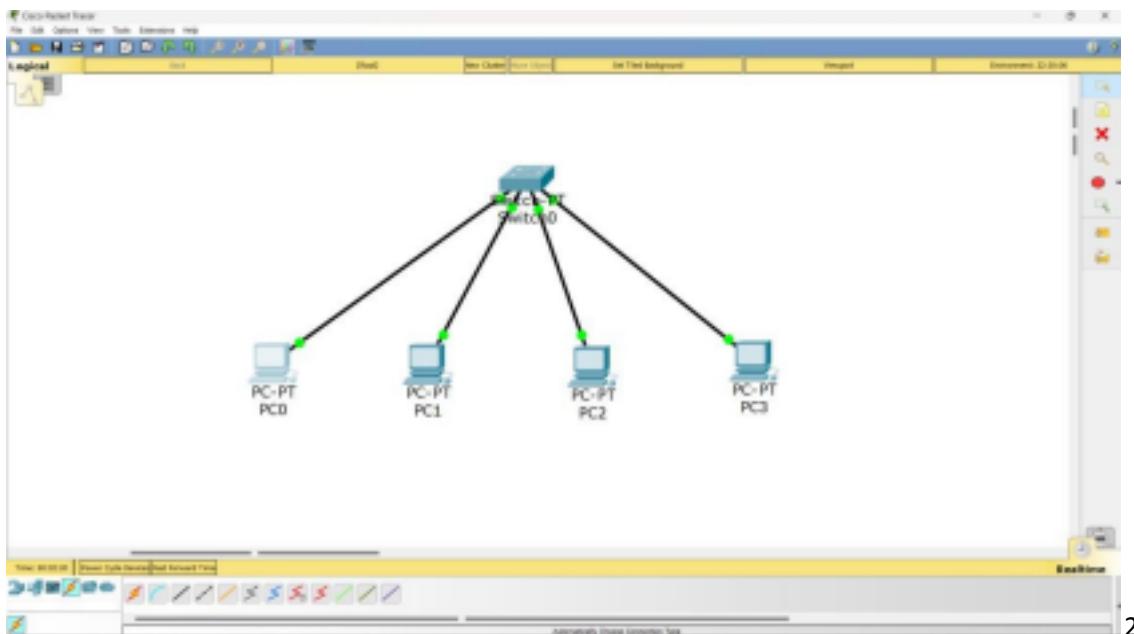
Pinging 10.0.0.2 with 32 bytes of data:

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

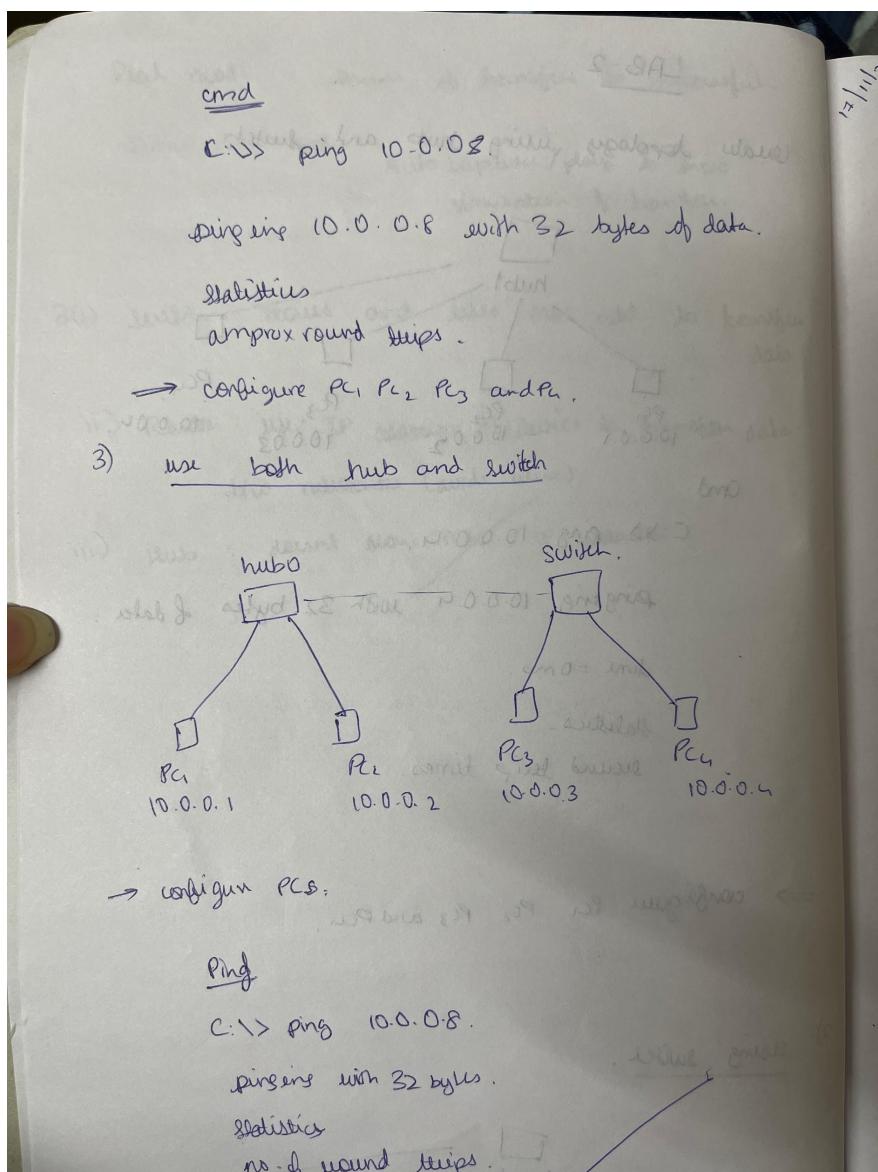
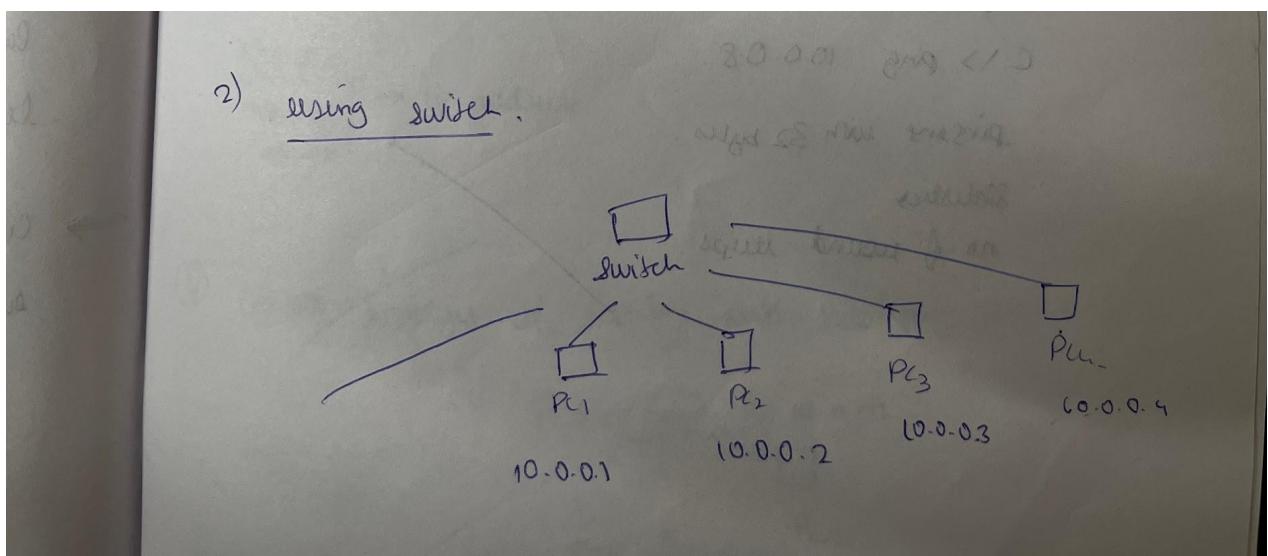
Ping statistics for 10.0.0.2:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 1ms

C:\>
```

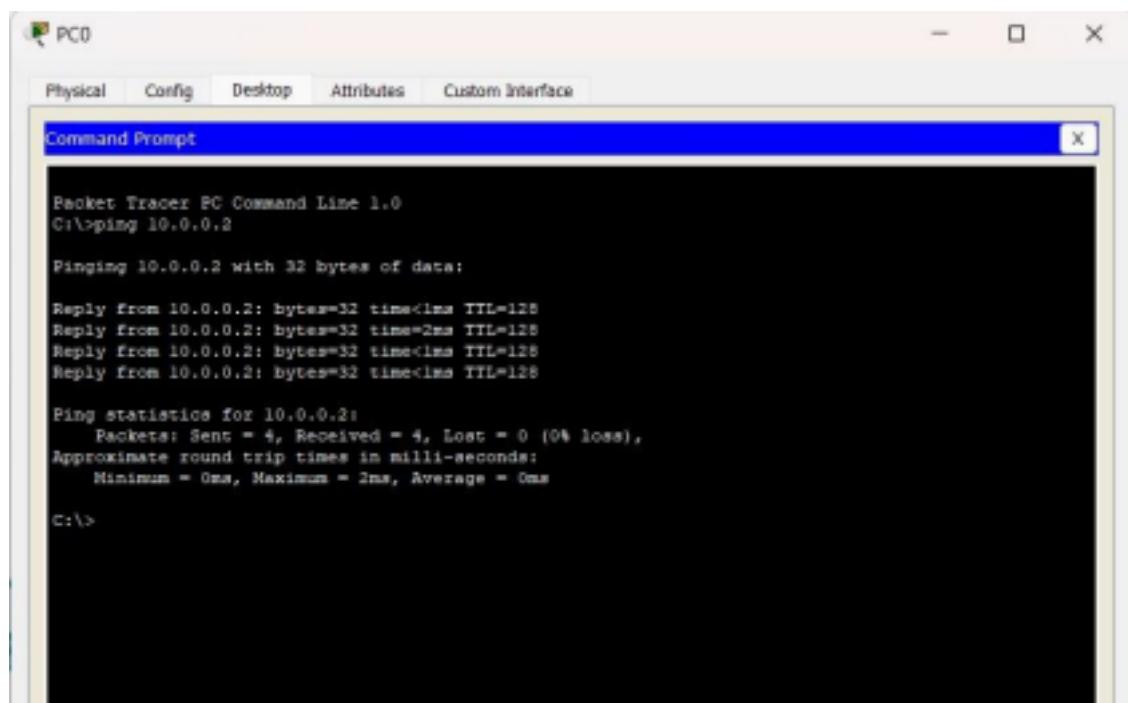
Switch Topology



Procedure



Output



PC0

Physical Config Desktop Attributes Custom Interface

Command Prompt

```
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

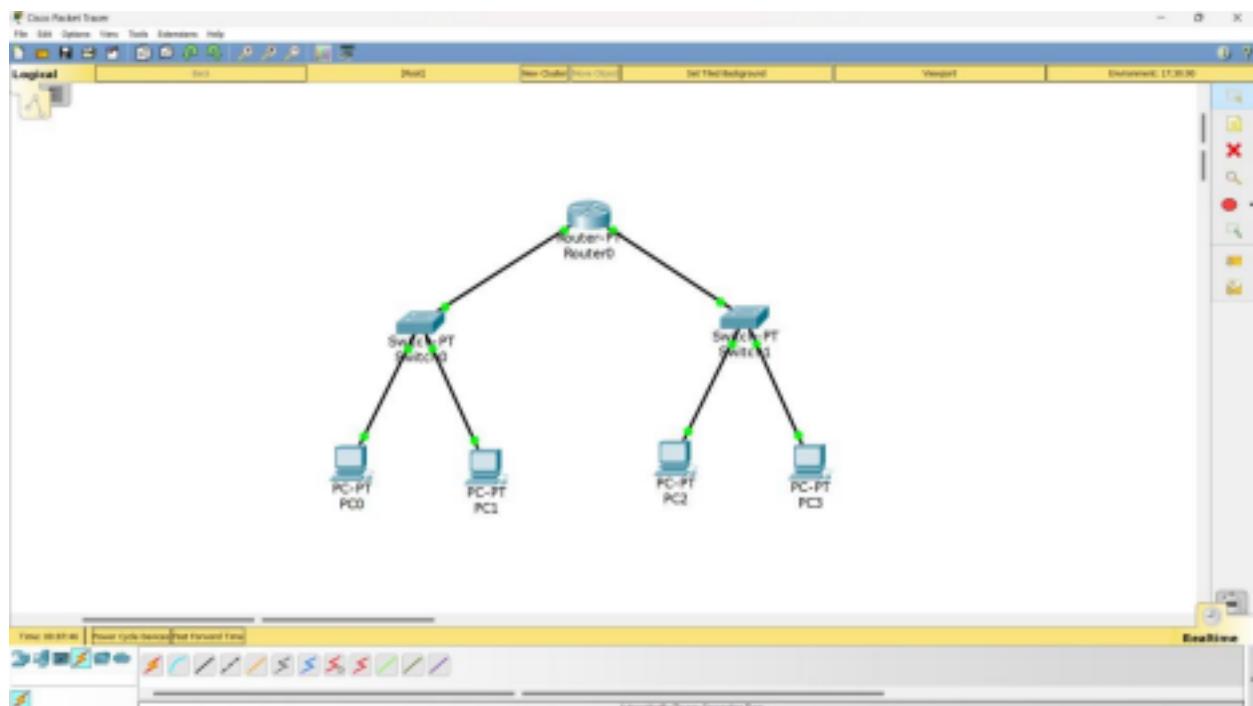
C:\>
```

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.

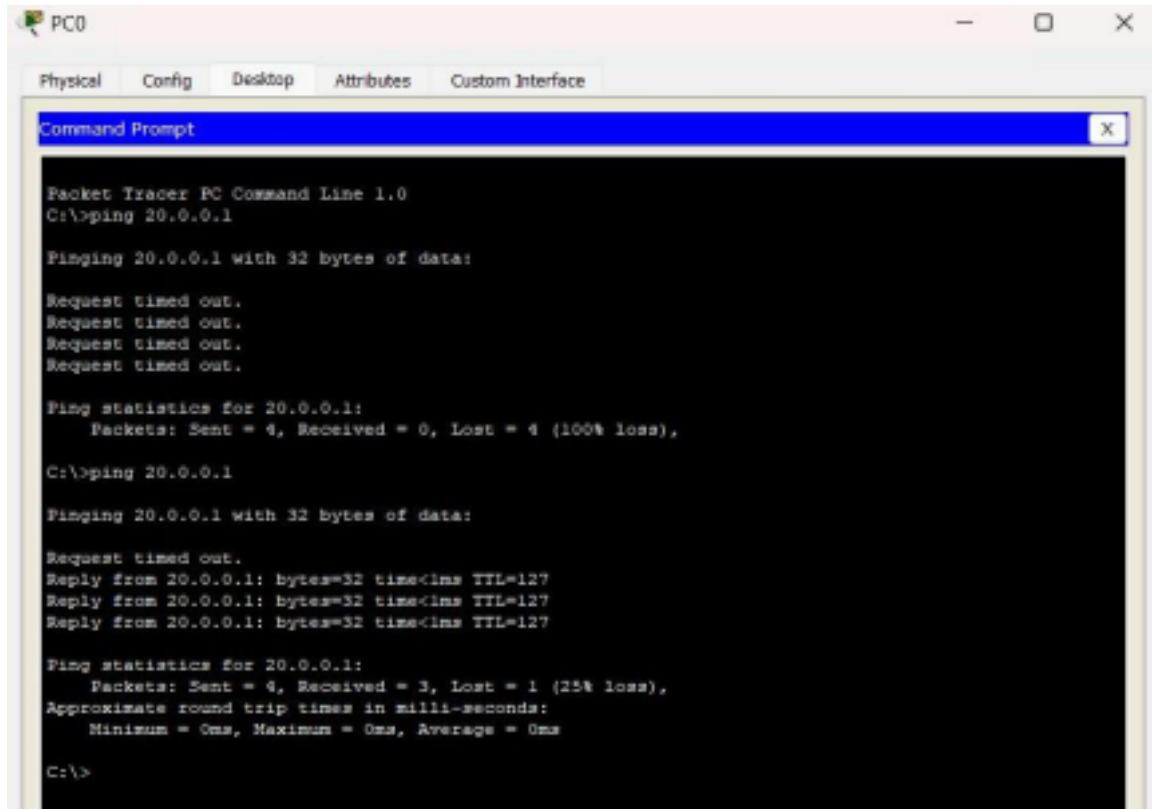
Topology



Procedure

```
Router>enable  
Router>configure terminal  
Enter configuration commands, one per line. End with CTRL/Z.  
Router(config)#interface FastEthernet0/0  
Router(config-if)#ip address 10.0.0.11 255.0.0.0  
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 10.0.0.12 255.0.0.0  
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)#
```

Output



The screenshot shows a Windows Command Prompt window titled "PC0". The window has a blue header bar with the title and standard window controls (minimize, maximize, close). Below the header is a menu bar with tabs: Physical, Config, Desktop, Attributes, and Custom Interface. The main area of the window is a black terminal-like interface displaying command-line output.

```
Packet Tracer PC Command Line 1.0
C:\pinging 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:\pinging 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

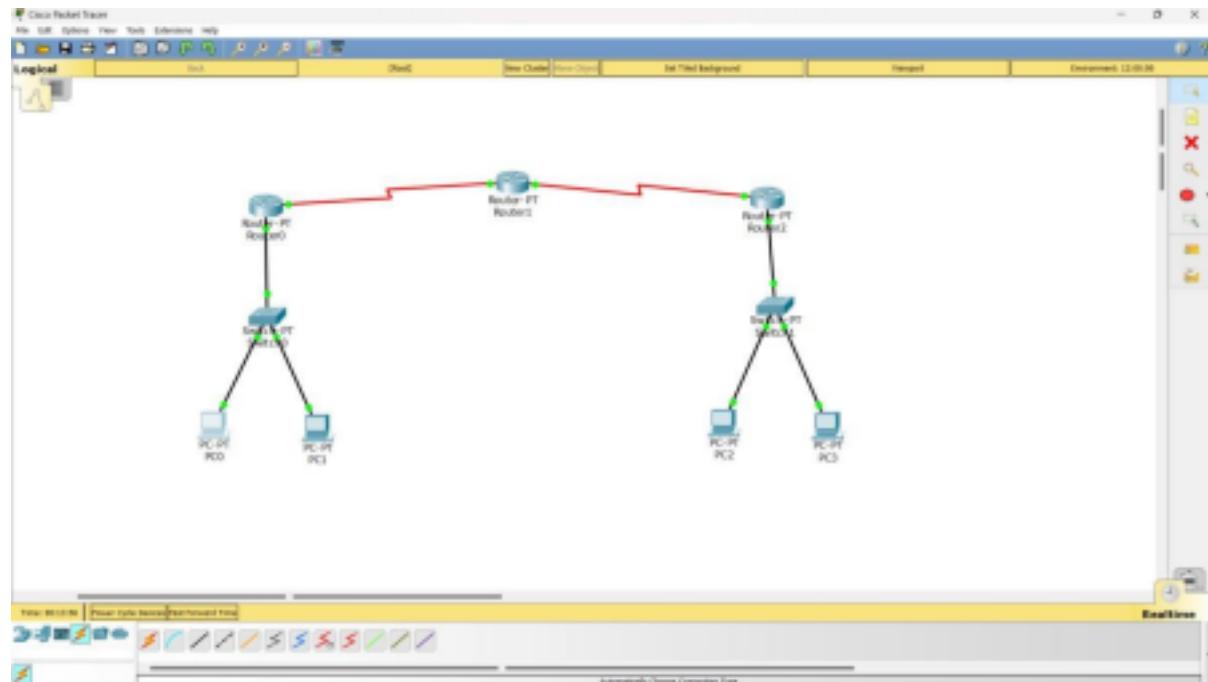
C:\>
```

Experiment No 3

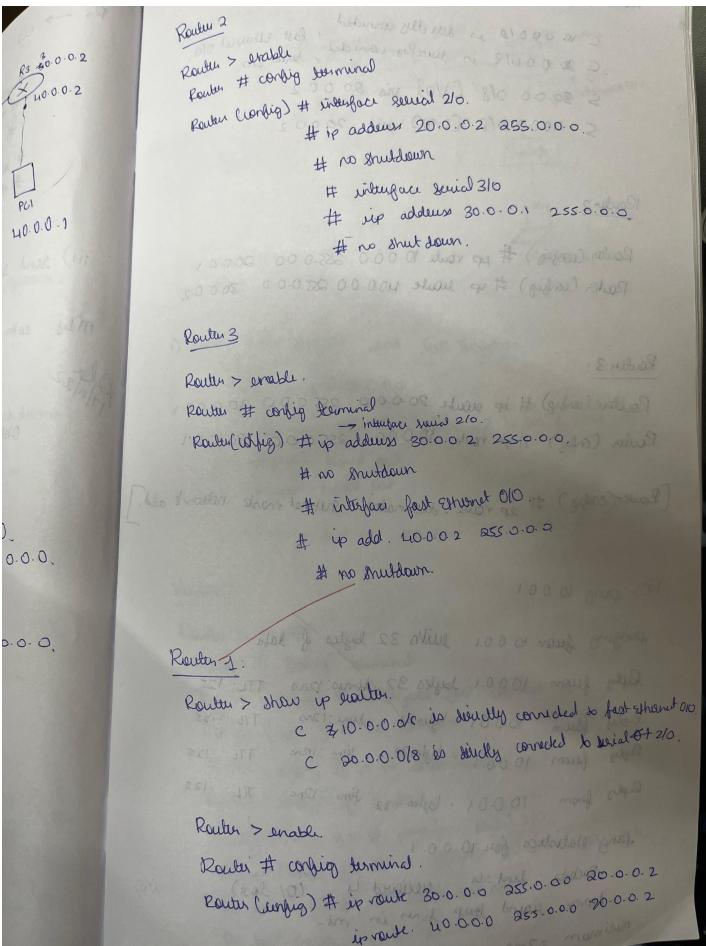
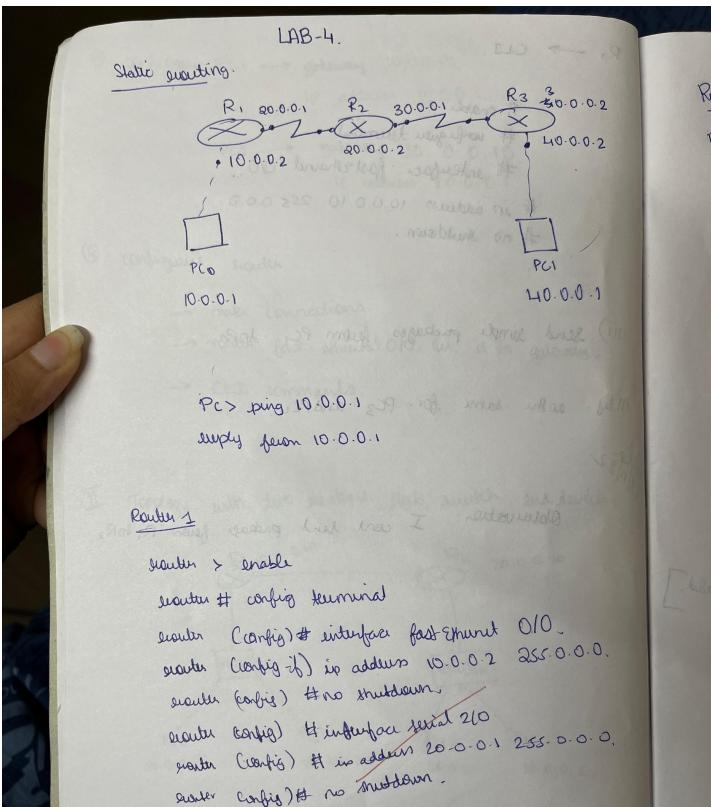
Aim of the program

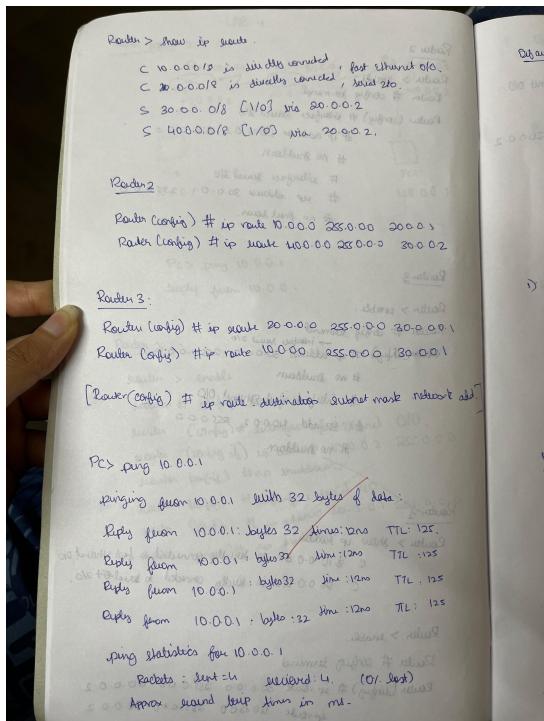
Configuring static and default route to the Router

Topology for static routing



Procedure





7

Output

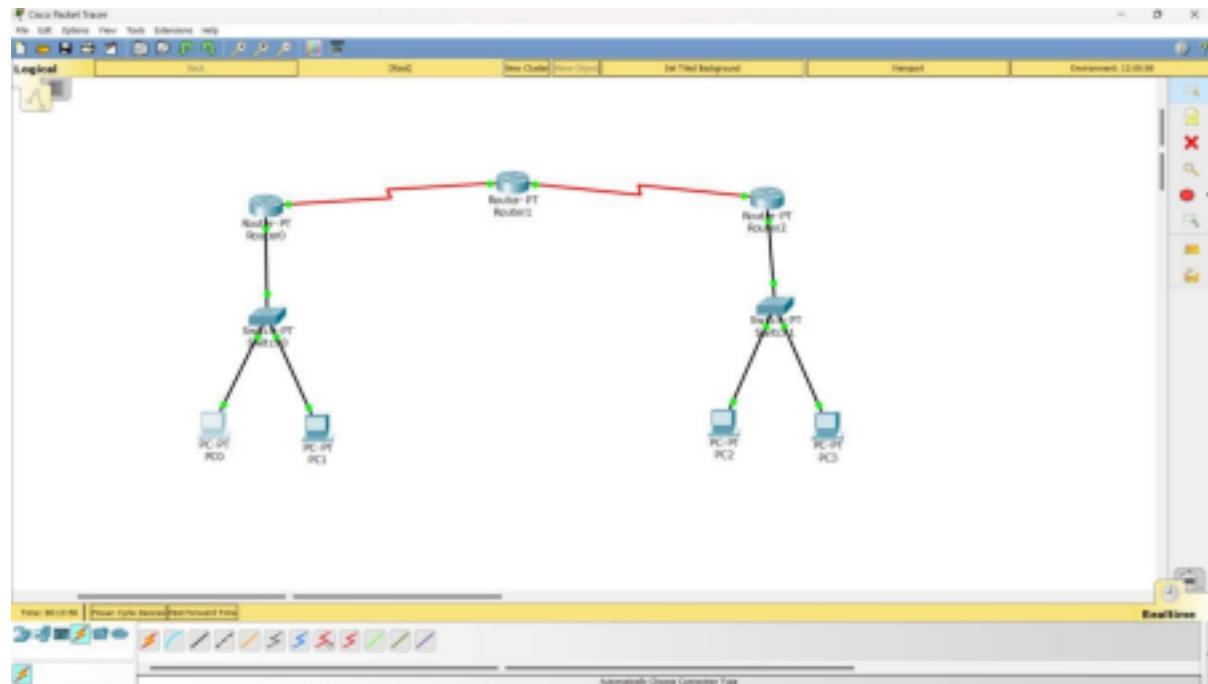
```
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

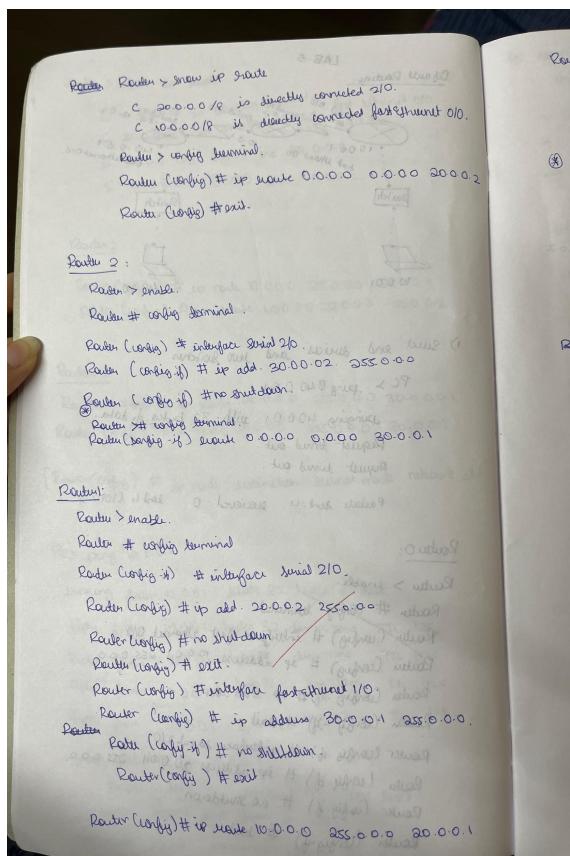
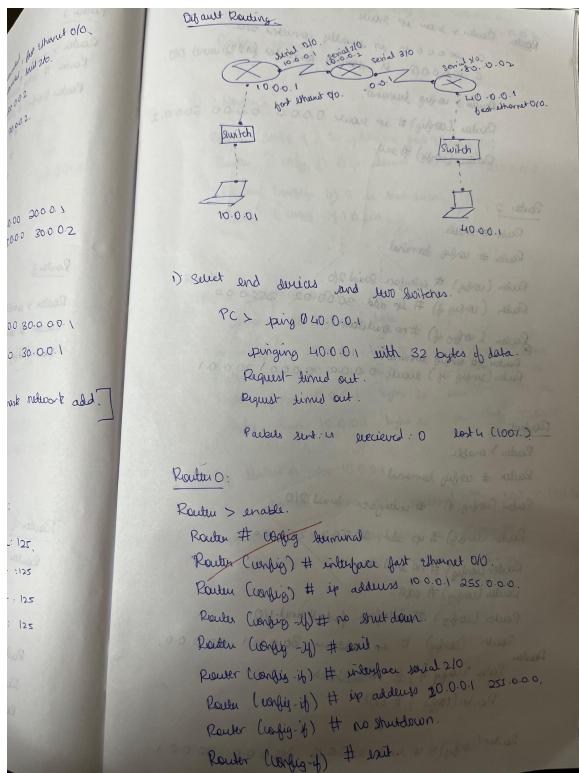
Reply from 40.0.0.1: bytes=32 time<1ms TTL=127

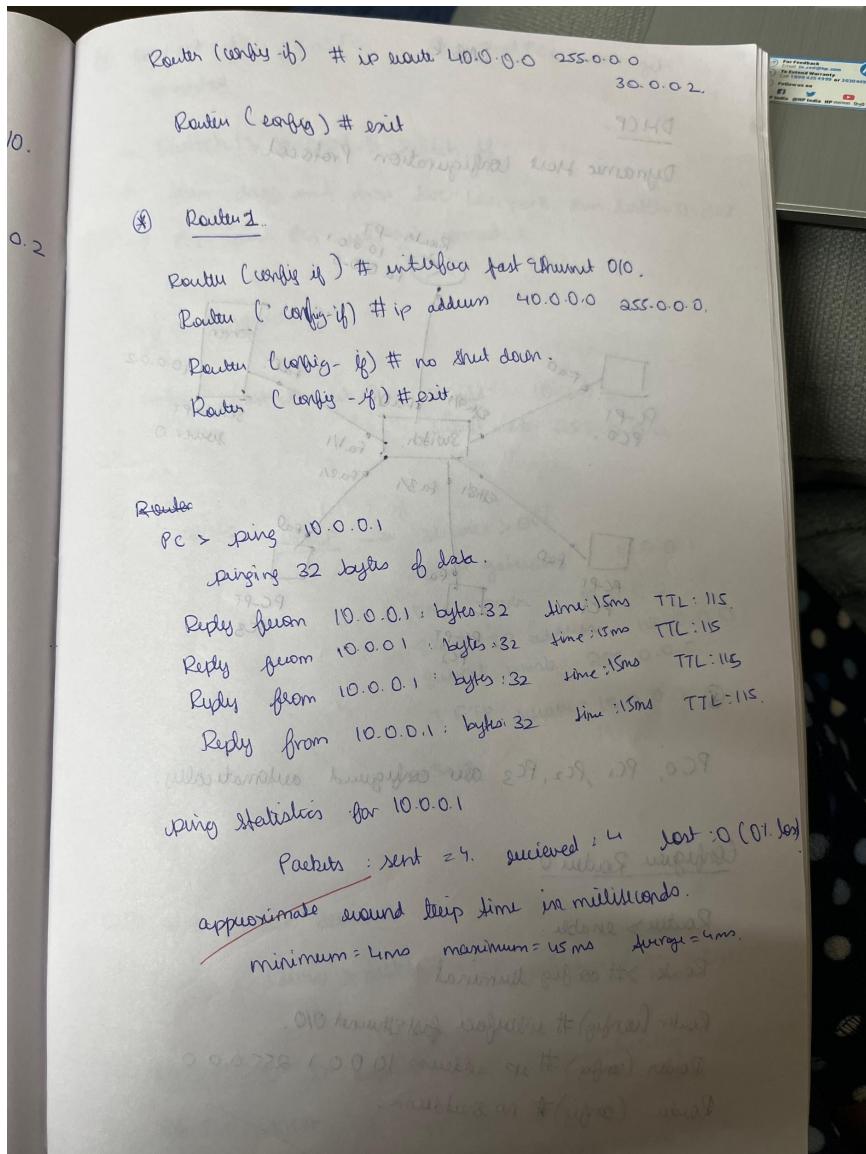
Ping statistics for 40.0.0.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

Topology for default routing



Procedure





Output

```

C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1: bytes=32 time<1ms TTL=127

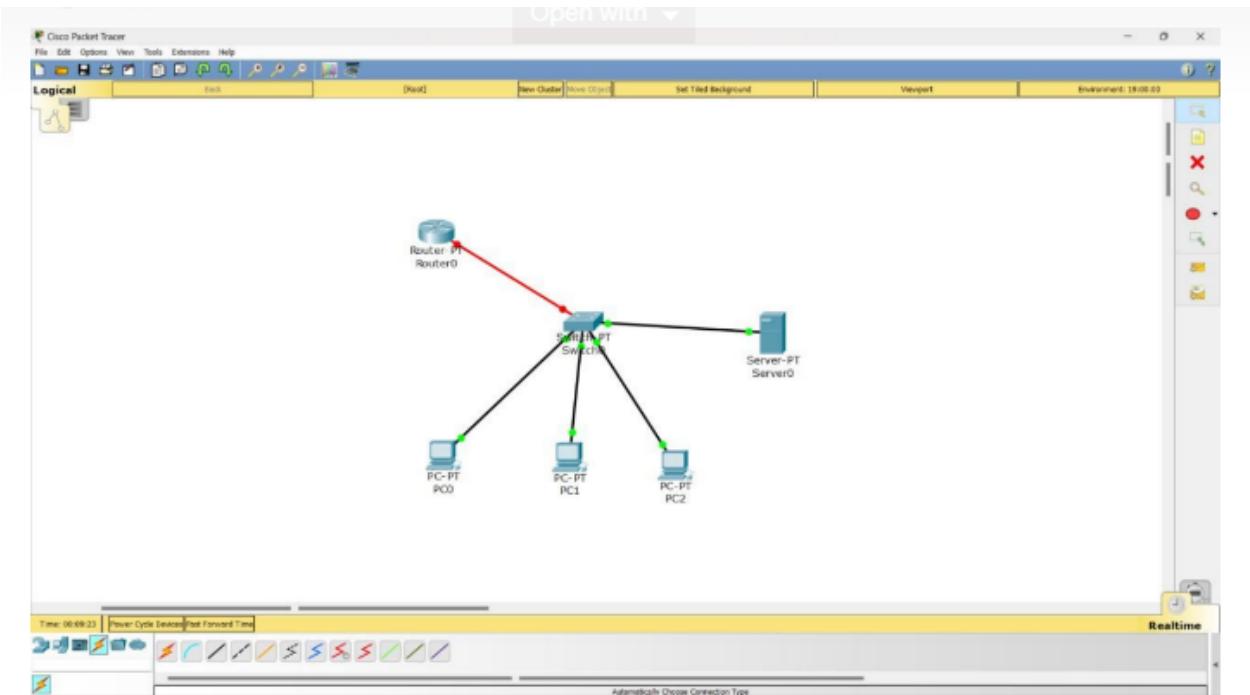
Ping statistics for 40.0.0.1:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
  Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
  
```

Experiment No 4

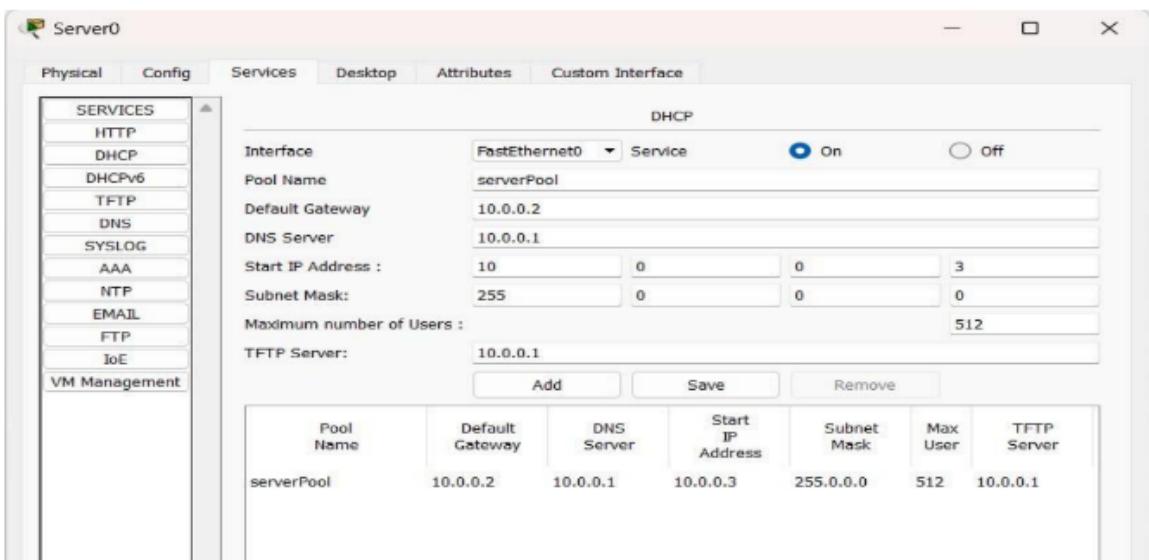
Aim of the program

Configuring DHCP within a LAN in a packet Tracer

Topology

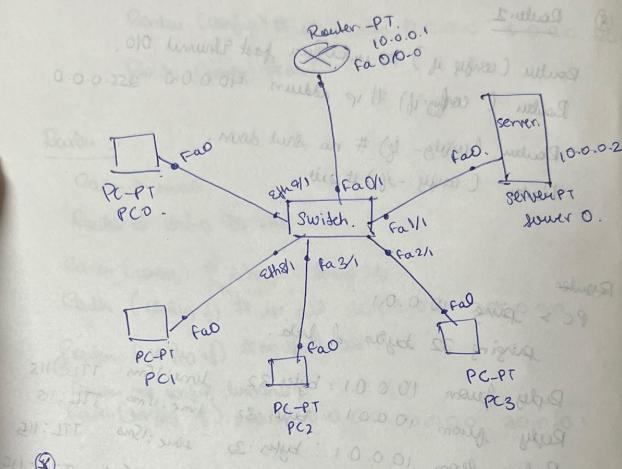


Procedure



DHCP.

Dynamic Host Configuration Protocol.



PC₀, PC₁, PC₂, PC₃ are configured automatically.

Configure Router 0.

Router > enable.

Router ># config terminal

Router (config)# interface fastethernet 0/0.

Router (config)# ip address 10.0.0.1 255.0.0.0

Router (config)# no shutdown.

① Connect

conn

→ Sub

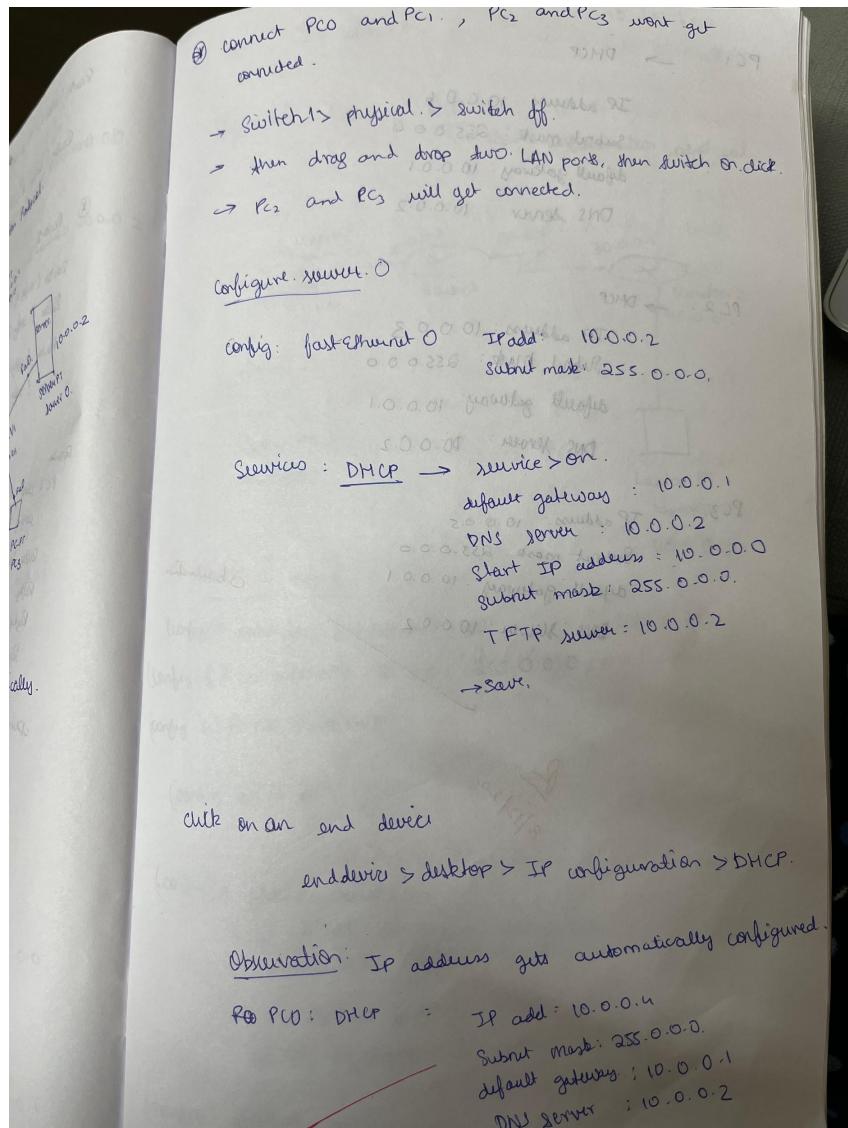
→ A

→ I

Config

Conn

Conn



PC1 → DHCP

IP address: 10.0.0.6

Subnet mask: 255.0.0.0

default gateway: 10.0.0.1

DNS server: 10.0.0.2

PC2: → DHCP

IP address: 10.0.0.3

Subnet mask: 255.0.0.0

default gateway: 10.0.0.1

DNS server: 10.0.0.2

PC3: → DHCP

IP address: 10.0.0.5

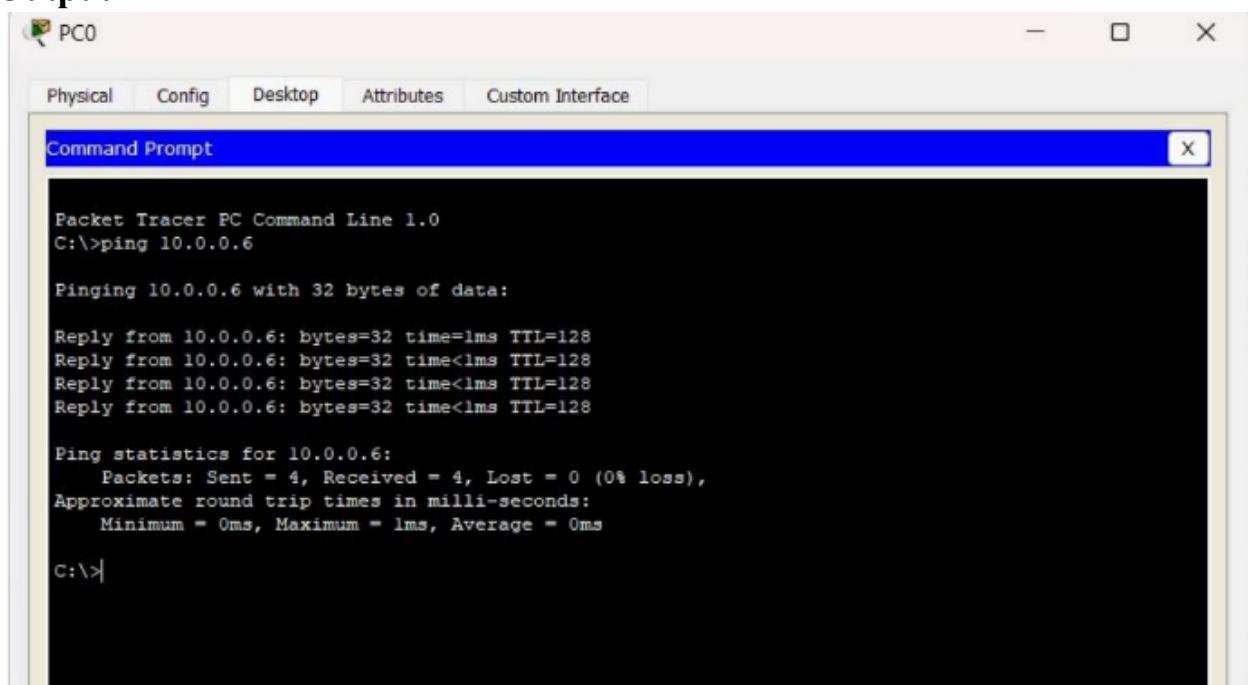
Subnet mask: 255.0.0.0

default gateway: 10.0.0.1

DNS server: 10.0.0.2

R
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

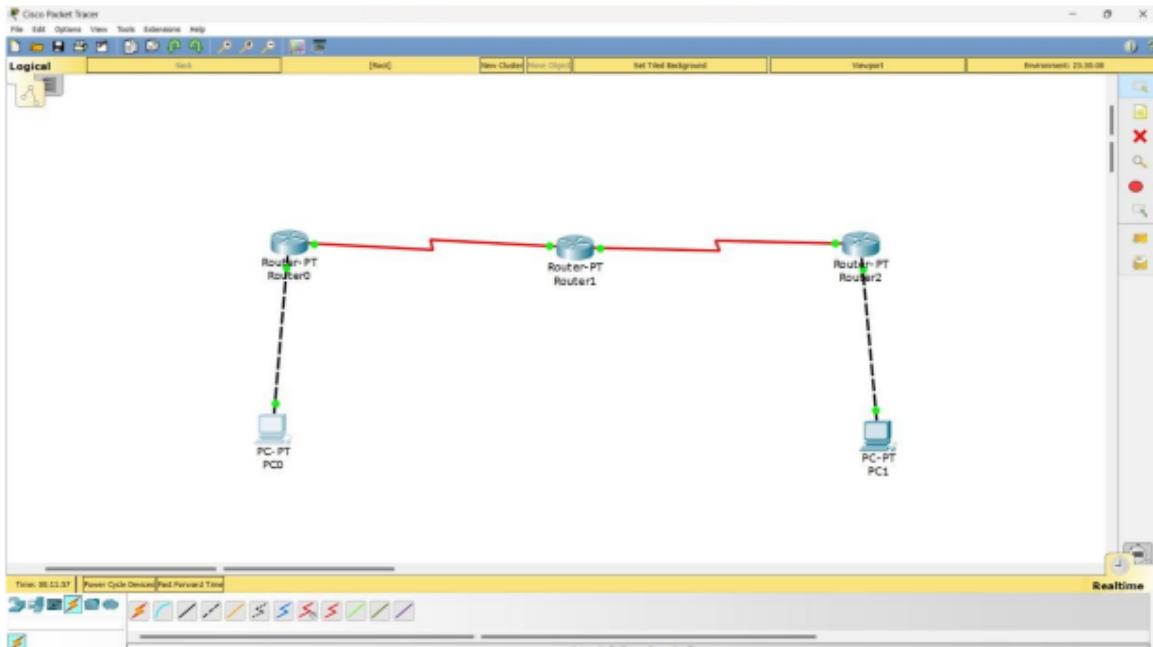
C:\>

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-if)#
Router(config-if)#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)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to down

Router(config-if)#
Router(config-if)#exit
Router(config-if)#
Router(config-if)#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)#
This command applies only to DCE interfaces
Router(config-if)#no shutdown

Router(config-if)#
%LINK-5-CHANGED: Interface Serial2/0, changed state to down

Router(config-if)#
Router(config-if)#exit
Router(config-if)#
Router(config-if)#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

Router(config-if)#
%LINK-5-CHANGED: Interface Serial3/0, changed state to down

Router(config-if)#
Router(config-if)#exit
Router(config-if)#
Router(config-if)#router rip
Router(config-router)#network 30.0.0.0
Router(config-router)#network 20.0.0.0
Router(config-router)#exit
Router(config)#
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

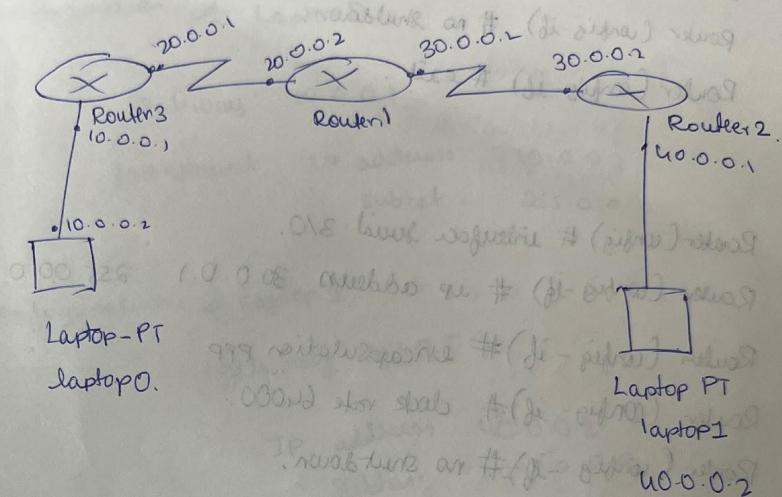
```

LAB - 7 .

15 | 12 | 22

Dynamic Routing

• Router information protocol



Router 1 config

```
(config)# interface fastEthernet 0/0
```

(config-if) # ip address 10.0.0.1 255.0.0.0

(config-if) # no shutdown

(config-if) # exit

~~(continues) The author is not available for further comments.~~

(config) # ~~interface serial 2/0~~ (sempre + 4 (se cifra) next

ip address 20.0.0.1 255.0.0.0

encapsulation ppp

clock scale 64000.

no shutdown.

no shutdown.

(config) # router eip 999 release 44

Router(config-router) #network 10.0.0.0

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

Router 1 configuration

```
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)# interface serial 3/0
```

```
Router(config)# 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)# router rip
```

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

```
Router(config-router)# network 30.0.0.0
```

Router 2 config

```
Router(config)# interface fastethernet 0/0
```

```
Router(config-if)# ip address 40.0.0.1 255.0.0.0
```

```
# no shutdown
```

```
Router(config)# interface serial 2/0
```

```
Router(config-if)# ip add. 30.0.0.2 255.0.0.0
```

```
# encapsulation PPP
```

```
# no shutdown
```

Router (config)

Router

Configuration

Gal

part

Configuration

Laptop 0:

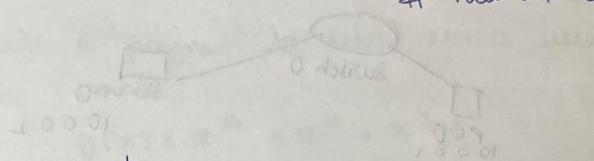
host 0:

Ping

Observation

Router (config) # Router config

Router (config router) # network 30.0.0.0
network 40.0.0.0



Configuration of Laptop D.

Gateway: 10.0.0.1

Interface: IP address 10.0.0.2
subnet: 255.0.0.0

Configuration of Laptop I:

Gateway: 40.0.0.1

IP address: 40.0.0.2
subnet: 255.0.0.0

Laptop 0: cmd.

ping 40.0.0.2

timeout

ping 40.0.0.2

pinging 40.0.0.2 with 32 bytes of data

Reply from 40.0.0.2: bytes=32 time=2ms

Reply from 40.0.0.2: bytes=32 time=11ms

Reply from 40.0.0.2: bytes=32 time=12ms

Reply from 40.0.0.2: bytes=32 time=2ms

Reply from 40.0.0.2: bytes=32 time=2ms

Ping statistics for 40.0.0.2

Packets: Sent=4 Received=4 Lost=0

Observation:

There is no need to give configuration for the PC separately because we are dynamically using rip.

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

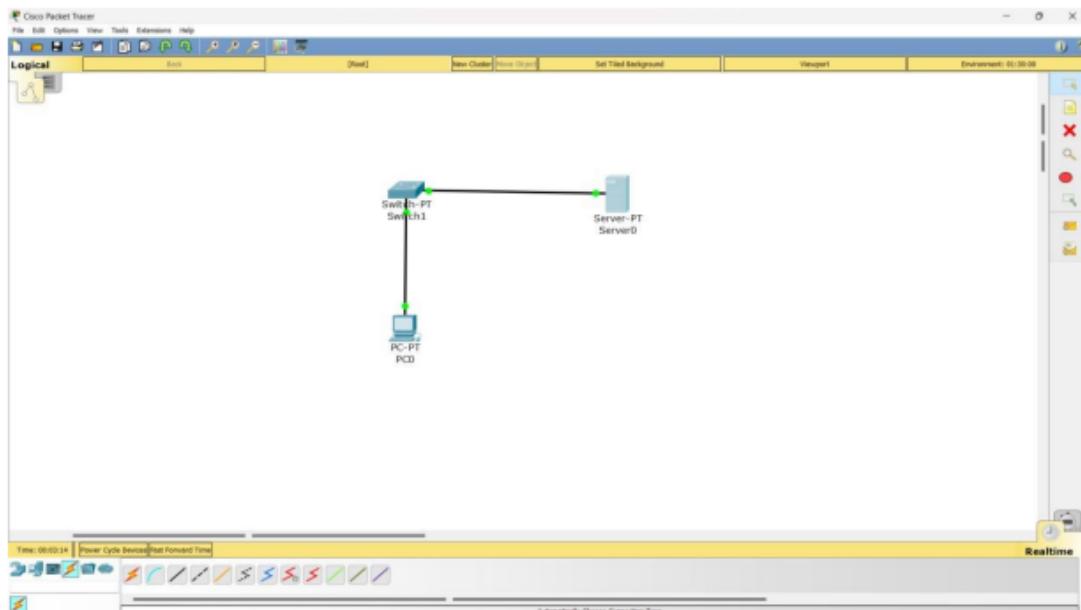
C:\>
```

Experiment No 6

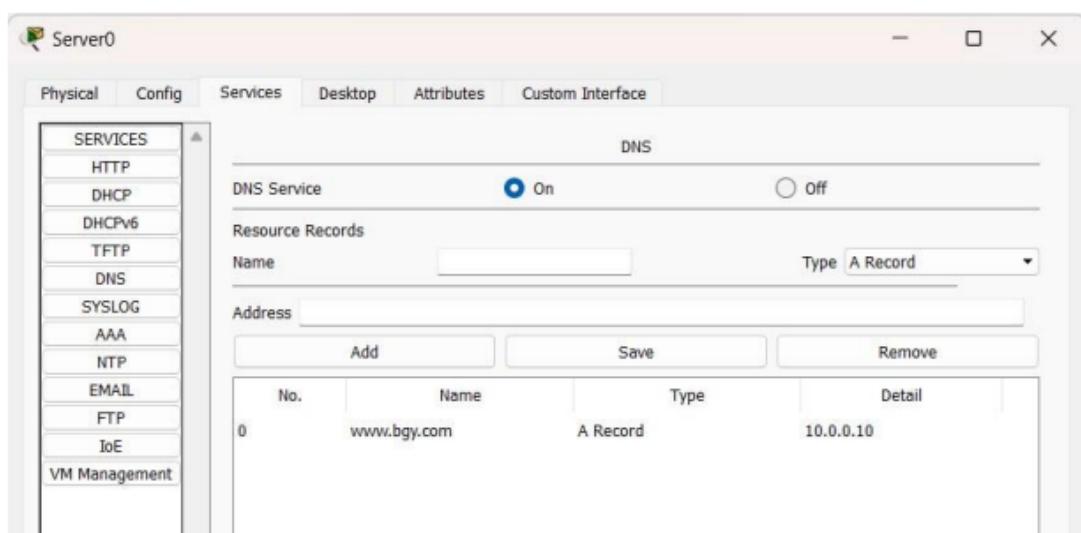
Aim of the program

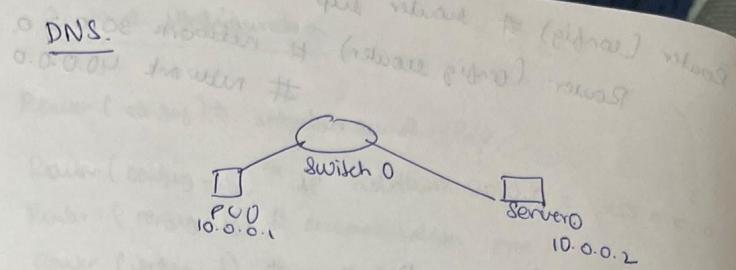
Demonstration of WEB server and DNS using Packet Tracer

Topology



Procedure





PC configuration.

IP address: 10.0.0.1
Subnet mask: 255.0.0.0

Gateway: 10.0.0.254

Server 0 configuration.

Services > DNS

Name: www.ankit.com
Address: 10.0.0.2

Add.

TFTP: on.

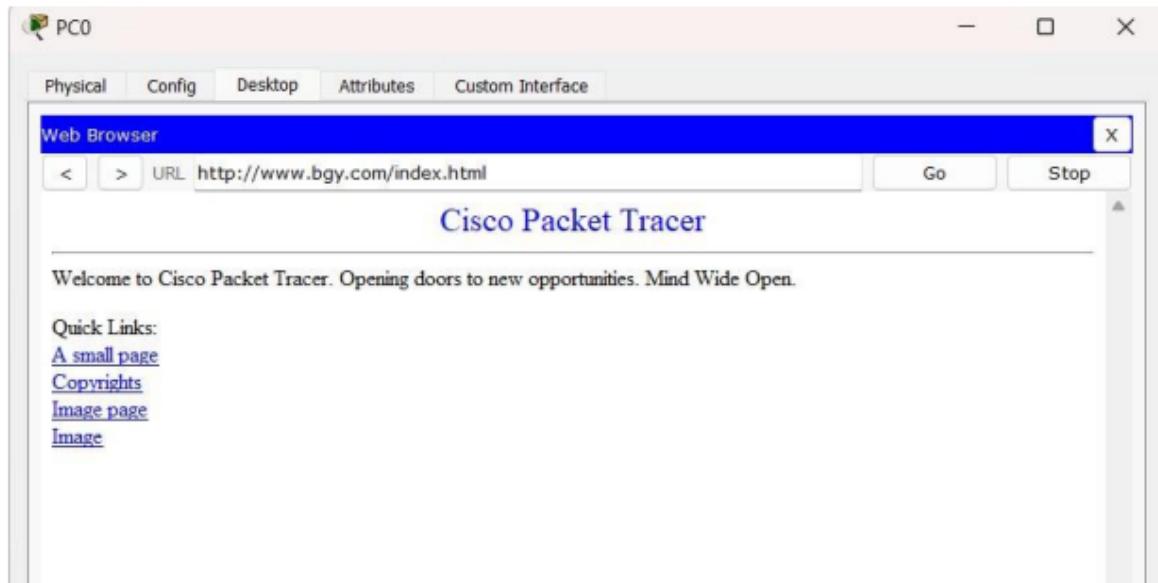
HTTP: file manager > helloworld.html.

edit → <html>
<h1> hello </h1>

PC

desktop: web browser: www.ankit.com

Output



Cycle-2

Experiment No 1

Aim of the Experiment

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

Code

```
import
java.util.*;

public class Main{
    public static int n;

    public static void main(String[] args){
        Scanner in=new Scanner(System.in);
        Main ob=new Main();

        String data,data_copy,zero="0000000000000000",ans,data_r;
        System.out.print("Enter the data to be transferred:");
        data=in.nextLine();
        data_copy=data;
        data+=zero;
        n=data_copy.length();

        System.out.println("Divisor:1000100000100001");

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

        System.out.println("CheckSum: "+data.substring(n));
        data_copy=data_copy.substring(0,n)+data.substring(n);
        System.out.println("Final Codeword: "+data_copy);

        System.out.print("Enter the data received at the destination:");
        data_r=in.nextLine();
        data_r=ob.divide(data_r);
        System.out.println("Remainder:"+data_r);

        zero="0000000000000000";
        if(data_r.equals(zero)==true){
            System.out.println("No error");
        }
    }
}
```

```
else{
    System.out.println("Error detected");
}

public String divide(String s){
    int i,j;
    char x;
    String div="10001000000100001";

    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

```
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>

    struct node
    {
        unsigned dist[20];
        unsigned from[20];
        unsigned hopcount[20];
    }rt[10];
    int main()
    {
        int costmat[20][20];
        int nodes,i,j,k,count=0;
        printf("\nEnter the number of routers : ");
        scanf("%d",&nodes);
        printf("\nEnter the cost matrix :\n");
        for(i=0;i<nodes;i++)
        {
            for(j=0;j<nodes;j++)
            {
                scanf("%d",&costmat[i][j]);
                if(costmat[i][j]>0){
                    rt[i].hopcount[j]=1;
                }
                else
                    rt[i].hopcount[j]=0;
                costmat[i][i]=0;
                rt[i].dist[j]=costmat[i][j];//initialise the distance equal to cost
                matrix
                rt[i].from[j]=j;
            }
        }
        do
        {
            count=0;
            for(i=0;i<nodes;i++)//We choose arbitrary vertex k and we
            calculate the direct distance from the node i to k using the cost
            matrix //and add the distance from k to node j
```

```
for(j=0;j<nodes;j++)
for(k=0;k<nodes;k++)

if(rt[i].dist[j]>costmat[i][k]+rt[k].dist[j])  { //We
calculate the minimum distance
rt[i].dist[j]=rt[i].dist[k]+rt[k].dist[j];
rt[i].hopcount[j]=rt[i].hopcount[k]+rt[k].hopcount[j]
;  rt[i].from[j]=k;
count++;
}
}while(count!=0);
for(i=0;i<nodes;i++)
{
printf("\n\n For router %d\n",i+1);
for(j=0;j<nodes;j++)
{
printf("\t\nnode %d via %d Distance %d
",j+1,rt[i].from[j]+1,rt[i].dist[j]);
printf("\tHop count:%d",rt[i].hopcount[j]);  }
}
printf("\n\n");
getch();
}
```

```

Enter the number of routers : 5

Enter the cost matrix :
0 1 2 -99 -99
1 0 -99 -99 -99
2 -99 0 3 4
-99 -99 3 0 -99
-99 -99 4 -99 0

For router 1

node 1 via 1 Distance 0      Hop count:0
node 2 via 2 Distance 1      Hop count:1
node 3 via 3 Distance 2      Hop count:1
node 4 via 3 Distance 5      Hop count:2
node 5 via 3 Distance 6      Hop count:2

For router 2

node 1 via 1 Distance 1      Hop count:1
node 2 via 2 Distance 0      Hop count:0
node 3 via 1 Distance 3      Hop count:2
node 4 via 1 Distance 6      Hop count:3
node 5 via 1 Distance 7      Hop count:3

For router 3

node 1 via 1 Distance 2      Hop count:1
node 2 via 1 Distance 3      Hop count:2
node 3 via 3 Distance 0      Hop count:0
node 4 via 4 Distance 3      Hop count:1
node 5 via 5 Distance 4      Hop count:1

For router 4

node 1 via 3 Distance 5      Hop count:2
node 2 via 3 Distance 6      Hop count:3
node 3 via 3 Distance 3      Hop count:1
node 4 via 4 Distance 0      Hop count:0
node 5 via 3 Distance 7      Hop count:2

For router 5

node 1 via 3 Distance 6      Hop count:2
node 2 via 3 Distance 7      Hop count:3
node 3 via 3 Distance 4      Hop count:1
node 4 via 3 Distance 7      Hop count:2
node 5 via 5 Distance 0      Hop count:0

```

Experiment No 3

Aim of the Experiment

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

Code

```
#include
<stdio.h>

#define INFINITY 9999
#define MAX 10

void Dijkstra(int Graph[MAX][MAX], int n, int start);

void Dijkstra(int Graph[MAX][MAX], int n, int start) {
    int cost[MAX][MAX], distance[MAX], pred[MAX];
    int visited[MAX], count, mindistance, nextnode, i, j;

    // Creating cost matrix
    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
            if (Graph[i][j] == 0)
                cost[i][j] = INFINITY;
            else
                cost[i][j] = Graph[i][j];

    for (i = 0; i < n; i++) {
        distance[i] = cost[start][i];
        pred[i] = start;
        visited[i] = 0;
    }

    distance[start] = 0;
    visited[start] = 1;
    count = 1;

    while (count < n - 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 + cost[nextnode][i] < distance[i]) {
distance[i] = mindistance + cost[nextnode][i];
pred[i] = nextnode;
}
count++;
}

for (i = 0; i < n; i++)
if (i != start) {
printf("\nDistance from source to %d: %d", i, distance[i]);
}
}

int main() {
int Graph[MAX][MAX], i, j, n, u;
printf("Enter number of vertices:");
scanf("%d",&n);
printf("Enter adjacency matrix:");
for(i=0;i<n;i++){
for(j=0;j<n;j++){
scanf("%d",&Graph[i][j]);
}
}
printf("Enter the starting vertex:");
scanf("%d",&u);
Dijkstra(Graph, n, u);

return 0;
}

Enter number of vertices:5
Enter adjacency matrix:0 1 2 0 0
1 0 0 0 0
2 0 0 3 4
0 0 3 0 0
0 0 4 0 0
Enter the starting vertex:0

Distance from source to 1: 1
Distance from source to 2: 2
Distance from source to 3: 5
Distance from source to 4: 6

...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>

#define bucketSize 500

void bucketInput(int a,int b)
{
    if(a > bucketSize)
        printf("\n\t\tBucket overflow");
    else{
        while(a > b){
            printf("\n\t\t%d bytes outputted.",b);
            a-=b;
        }
        if(a > 0)
            printf("\n\t\tLast %d bytes sent\t",a);
        printf("\n\t\tBucket output successful");
    }
}

int main()
{
    int op,pktSize;
    printf("Enter output rate : ");
    scanf("%d",&op);
    for(int i=1;i<=5;i++)
    {
        pktSize=rand()%700;
        printf("\nPacket no %d \tPacket size = %d",i,pktSize);
        bucketInput(pktSize,op);
    }
    return 0;
}
```

OUTPUT:

```
Enter output rate : 400

Packet no 1      Packet size = 183
                  Last 183 bytes sent
                  Bucket output successful
Packet no 2      Packet size = 186
                  Last 186 bytes sent
                  Bucket output successful
Packet no 3      Packet size = 177
                  Last 177 bytes sent
                  Bucket output successful
Packet no 4      Packet size = 215
                  Last 215 bytes sent
                  Bucket output successful
Packet no 5      Packet size = 393
                  Last 393 bytes sent
                  Bucket output successful

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

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

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()
```

25

Output

```
C:\Users\Bhargava\Downloads>python clitcp.py
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 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

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

```
C:\Users\Bhargava\Downloads>python cliudp.py
Enter file name main.cpp
From Server: b'#include <bits/stdc++.h>
using namespace std;
class Node {
public:
    int key;
    Node *left, *right, *parent;
    bool color = 0; // 1 -> black; 0 -> red
};

void inorderTraversal(Node *head) {
    if (head != NULL) {
        inorderTraversal(head->left);
        cout << head->key << " ";
        inorderTraversal(head->right);
    }
}

Node *leftRotate(Node *x) {
    Node *y = x->right;
    Node *t = y->left;
    if (y->right != NULL)
        y->right->parent = x;
    if (x->parent == NULL)
        y->parent = NULL;
    else if (x == y->parent->left)
        y->parent->left = y;
    else
        y->parent->right = y;
    y->left = x;
    x->parent = y;
    return y;
}

Node *rightRotate(Node *y) {
    Node *x = y->left;
    Node *t = y->right;
    if (y->left != NULL)
        y->left->parent = y;
    if (x->parent == NULL)
        y->parent = NULL;
    else if (x == y->parent->right)
        y->parent->right = y;
    else
        y->parent->left = y;
    x->parent = y;
    return x;
}

Node *bstInsert(Node *head, int val) {
    Node *newNode = new Node(val);
    if (head == NULL)
        head = newNode;
    else {
        Node *curr = head;
        Node *prev = NULL;
        while (curr != NULL) {
            if (val < curr->key) {
                curr = curr->left;
            } else {
                curr = curr->right;
            }
            if (val < prev->key)
                prev->right = newNode;
            else
                prev->left = newNode;
            prev = curr;
        }
    }
    return head;
}

int main() {
    Node *head = NULL;
    int k;
    cout << "Enter the number of elements: ";
    cin >> n;
    cout << "Enter the elements: ";
    for (int i = 0; i < n; i++) {
        cin >> k;
        head = bstInsert(head, k);
    }
    leftRotate(head);
    inorderTraversal(head);
    cout << endl;
}
```