VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT on

COMPUTER NETWORKS

Submitted by

SYED AYMAN AHMED (1BM20CS168)

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 **SYED AYMAN AHMED(1BM20CS168),** 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.

REKHA GS
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	17/11/22	Creating a topology and simulate sending a simple PDU from source to destination using hub and switch as connecting devices.
2	17/11/22	Configuring IP address to Routers in Packet Tracer. Exploring the following messages: Ping Responses, Destination unreachable, Request timed out, Reply
3	1/12/22	Configuring static and default route to the Router
4	8/12/22	Configuring DHCP within a LAN in a packet Tracer Configuring RIP
5	15/12/22	Routing Protocol in Routers
6	15/12/22	Demonstration of WEB server and DNS using Packet Tracer
7	29/12/22	Write a program for error detecting code using CRC-CCITT (16-bits).
8	6/1/23	Write a program for distance vector algorithm to find suitable path for transmission.
9	6/1/23	Implement Dijkstra's algorithm to compute the shortest path for a given topology.
10	13/1/23	Write a program for congestion control using leaky bucket algorithm.
11	20/1/23	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	20/1/23	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.

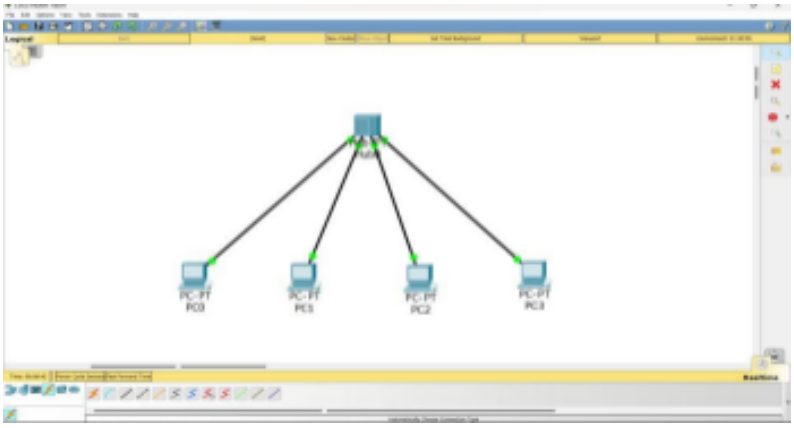
Cycle-1

Experiment - 1

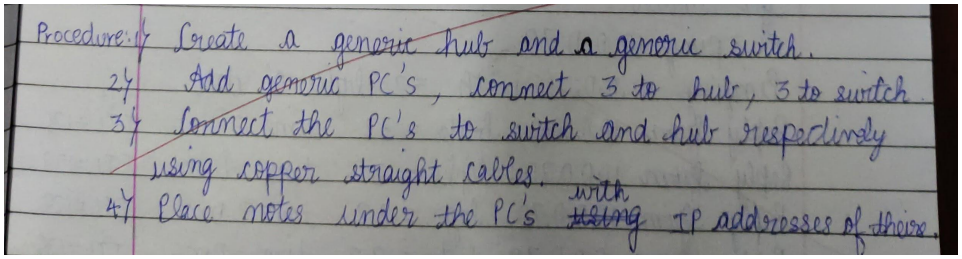
Aim of the program

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

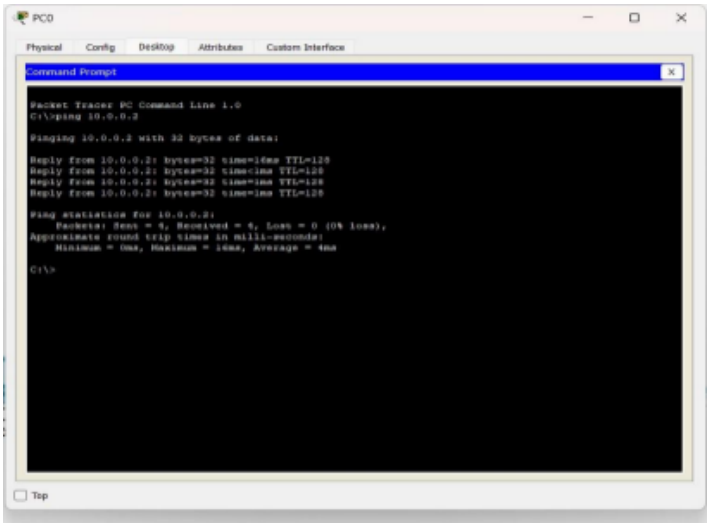
Hub Topology



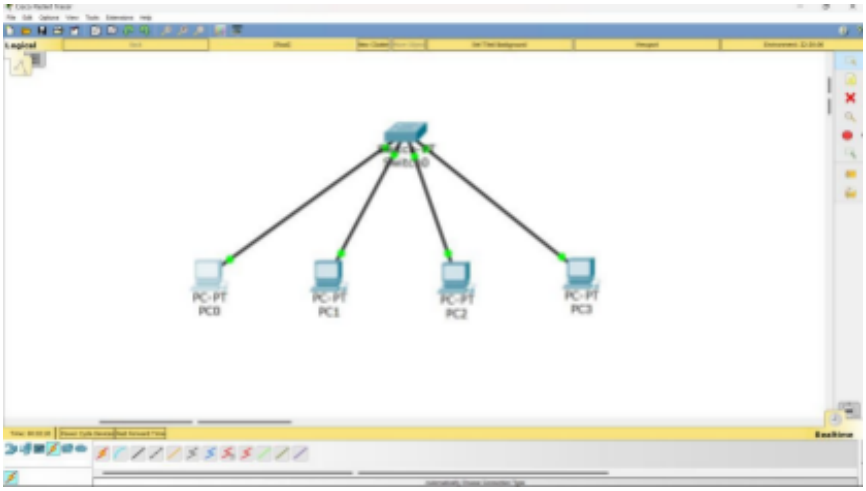
Procedure



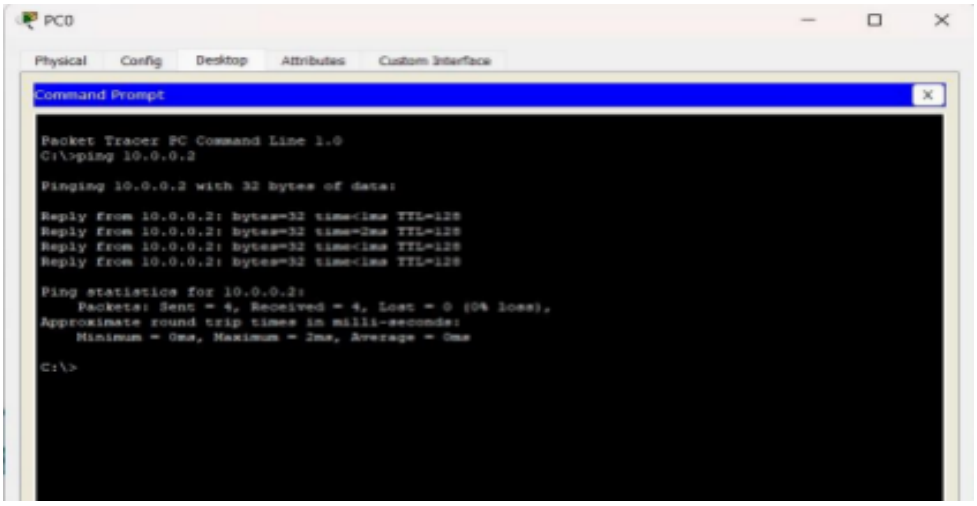
Output



Switch Topology



Output

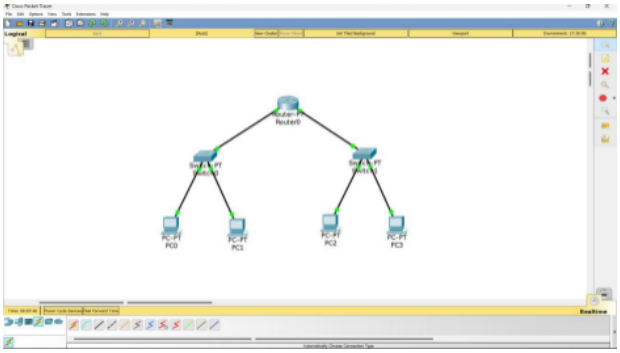


Experiment - 2

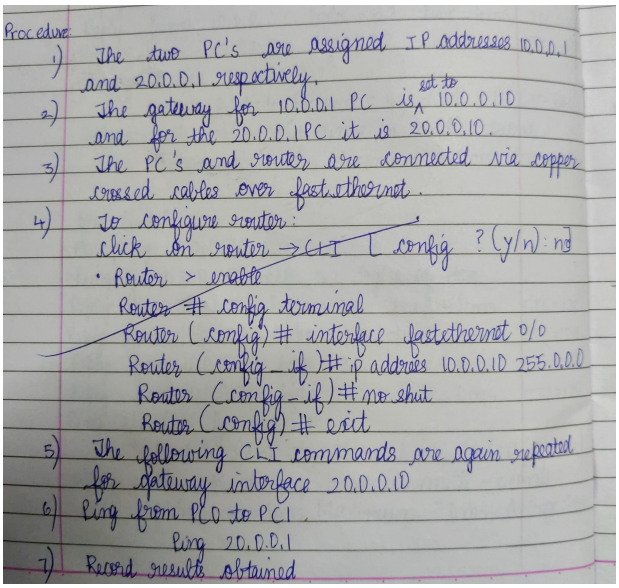
Aim of the program

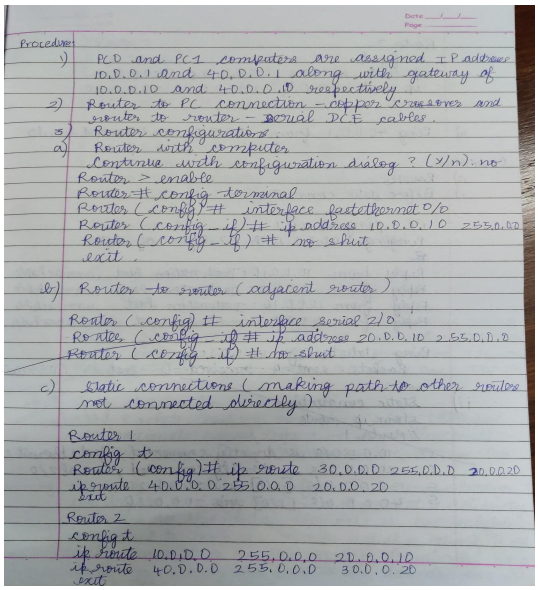
Configuring IP address to Routers in Packet Tracer. Exploring the following messages: Ping Responses, Destination unreachable, Request timed out, Reply.

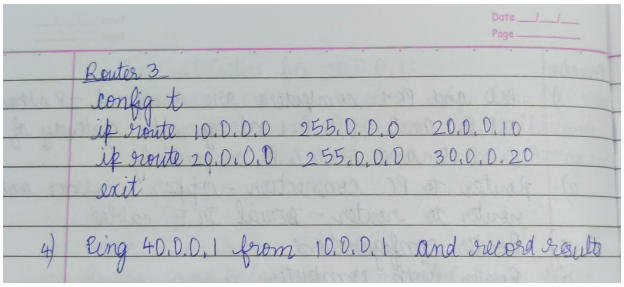
Topology



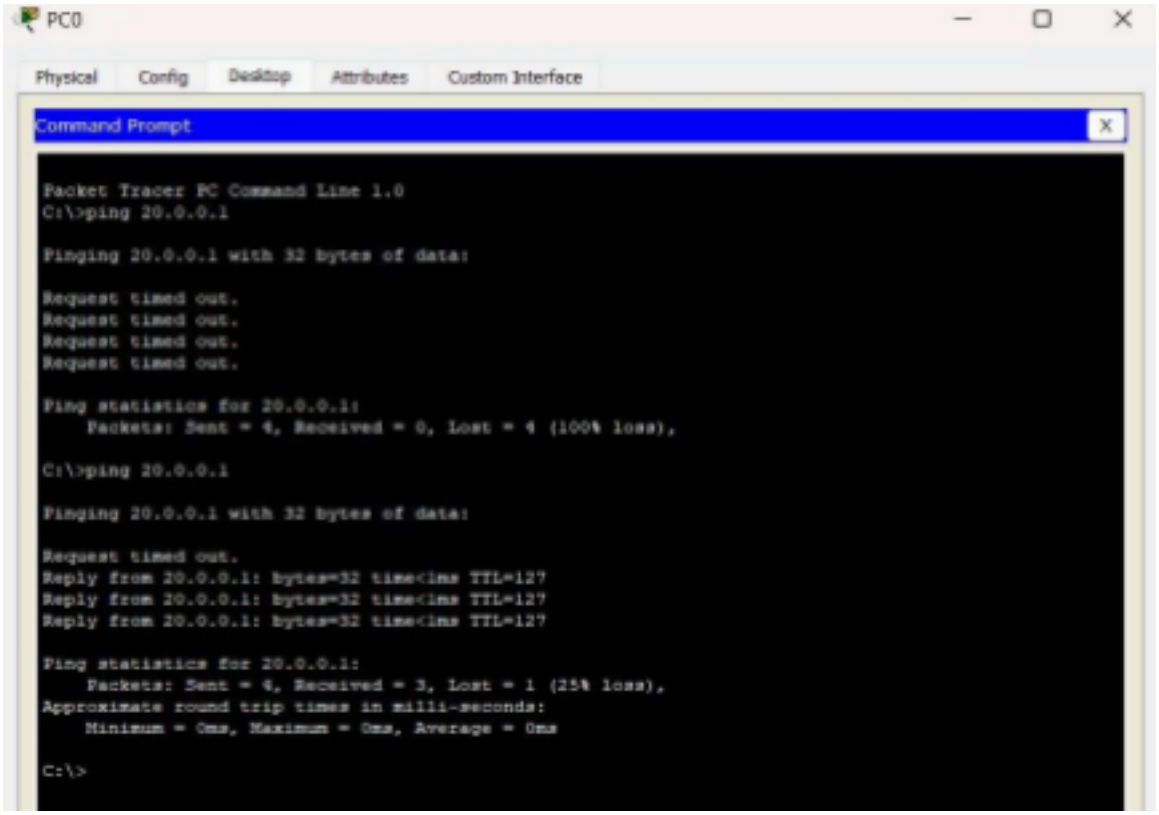
Procedure







Output

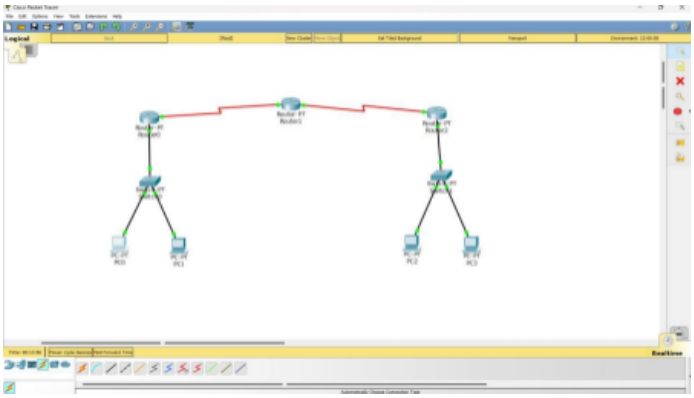

PC0
Physical
Config
Desktop
Attributes
Custom Interface
Command Prompt
Packet Tracer PC Command Line 1.0
C:\>ping 20.0.0.1
Pinging 20.0.0.1 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Ping statistics for 20.0.0.1:
Packets: Sent = 4, Received = 0, 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 - 3

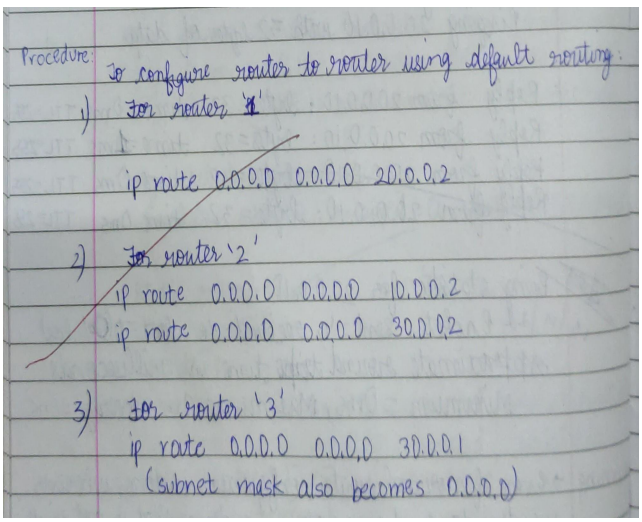
Aim of the program

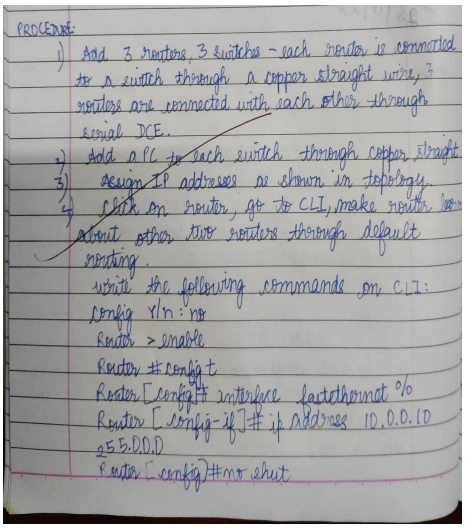
Configuring static and default route to the Router

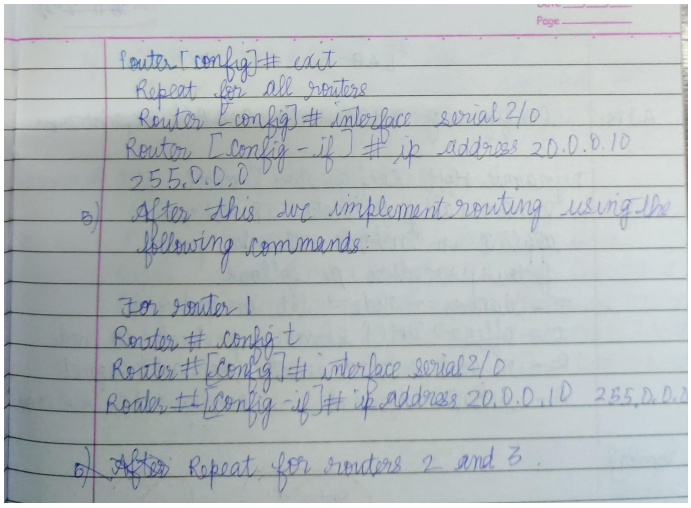
Topology for static routing



Procedure



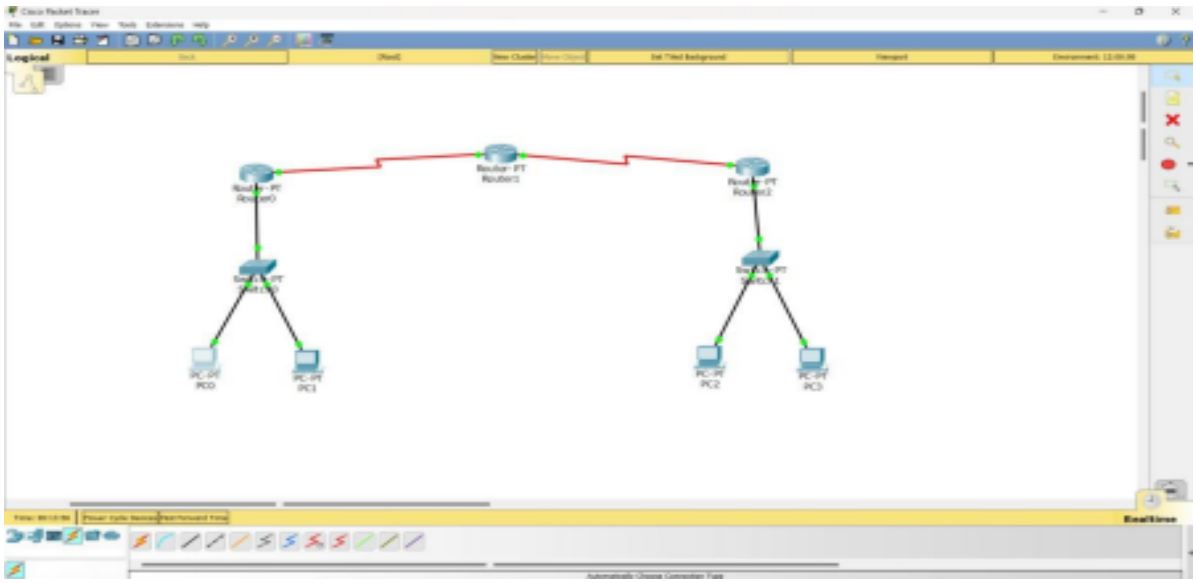




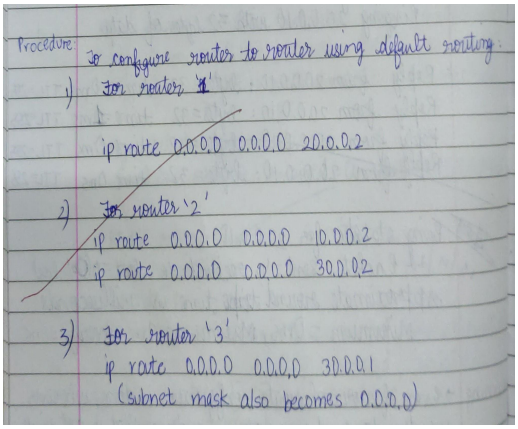
Output

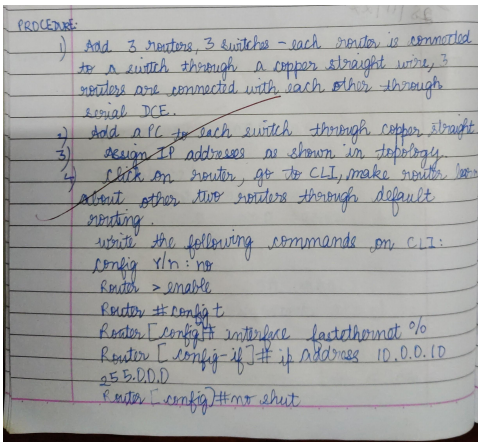
```
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

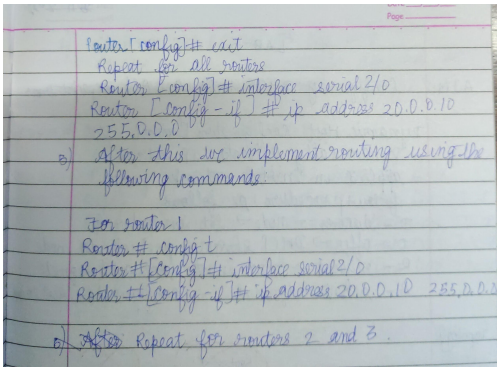
Topology for default routing



Procedure


Procedure
To configure router to router using default routing:
1) For router '1'
ip route 0.0.0.0 0.0.0.0 20.0.0.2
2) For router '2'
ip route 0.0.0.0 0.0.0.0 10.0.0.2
ip route 0.0.0.0 0.0.0.0 30.0.0.2
3) For router '3'
ip route 0.0.0.0 0.0.0.0 30.0.0.1
(subnet mask also becomes 0.0.0.0)


PROCEDURE:
1) Add 3 routers, 3 switches - each router is connected to a switch through a copper straight wire, 3 routers are connected with each other through serial DCE.
2) Add a PC to each switch through copper straight
3) Design IP addresses as shown in topology
4) Click on router, go to CLI, make router know about other two routers through default routing.
write the following commands on CLI:
config t/n : no
Router > enable
Router # config t
Router [config]# interface fastethernet 0/0
Router [config-if]# ip address 10.0.0.10 255.0.0.0
Router [config]# no shut


Router [config]# exit
Repeat for all routers
Router [config]# interface serial 2/0
Router [config-if]# ip address 20.0.0.10 255.0.0.0
3) After this we implement routing using the following commands:
For router 1
Router # config t
Router #[config]# interface serial 2/0
Router #[config-if]# ip address 20.0.0.10 255.0.0.0
5) After Repeat for routers 2 and 3.

Output

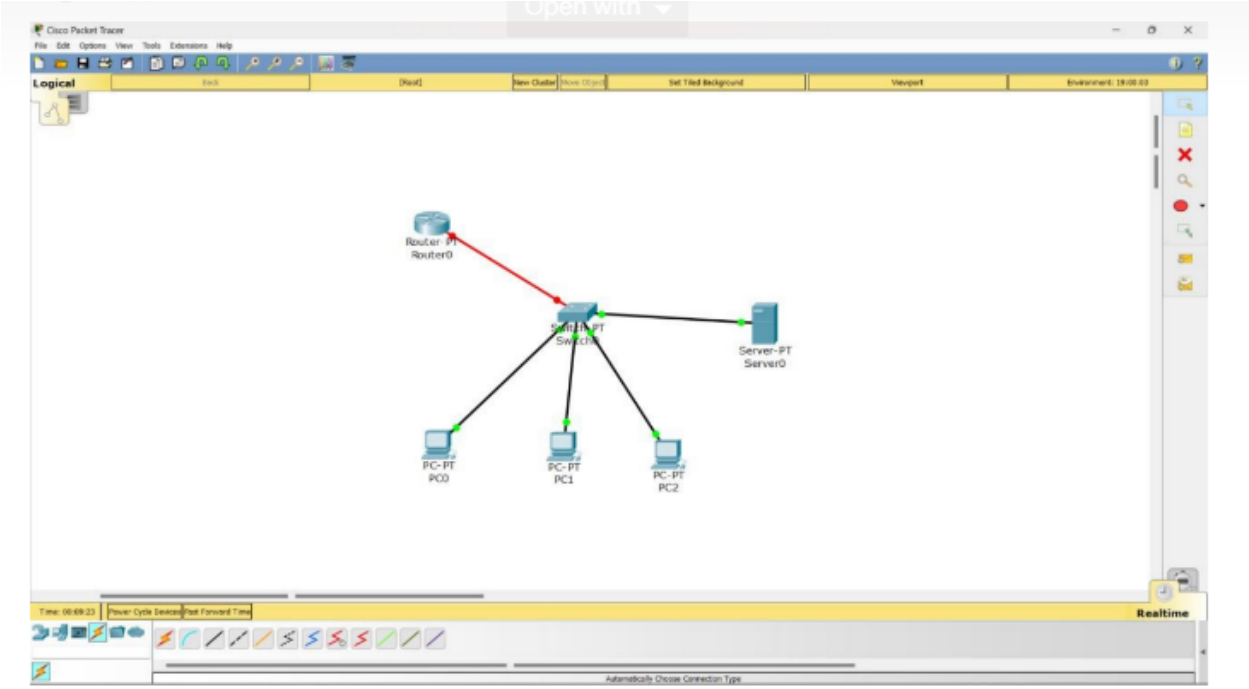
```
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

Reply from 40.0.0.1: bytes=32 time<1ms TTL=127
Reply from 40.0.0.1: bytes=32 time<1ms TTL=127
Reply from 40.0.0.1: bytes=32 time<1ms TTL=127
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 - 4

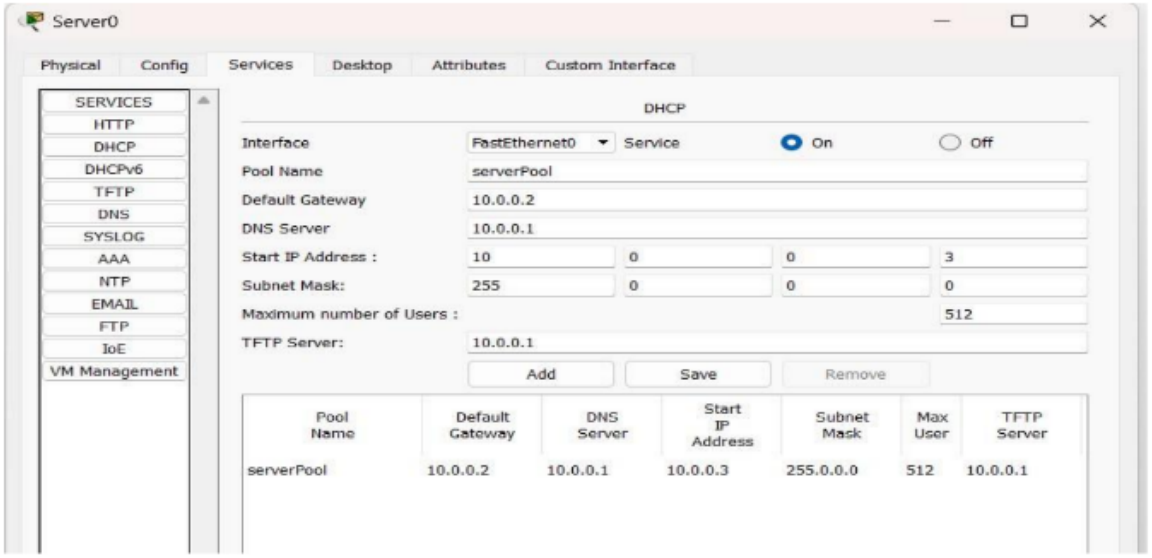
Aim of the program

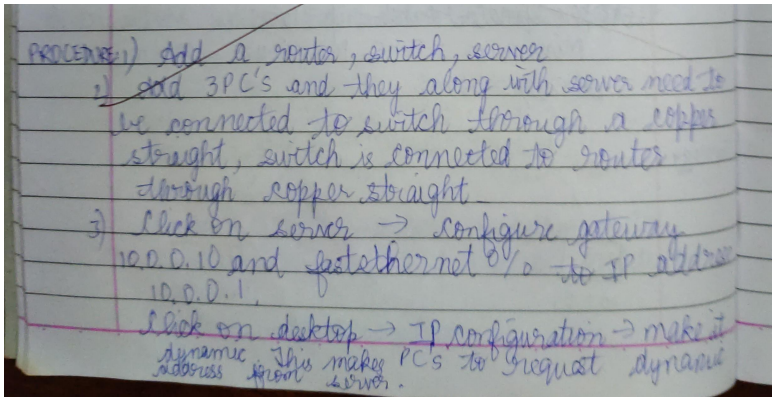
Configuring DHCP within a LAN in a packet Tracer

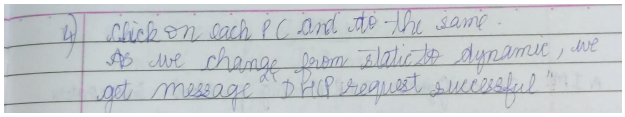
Topology



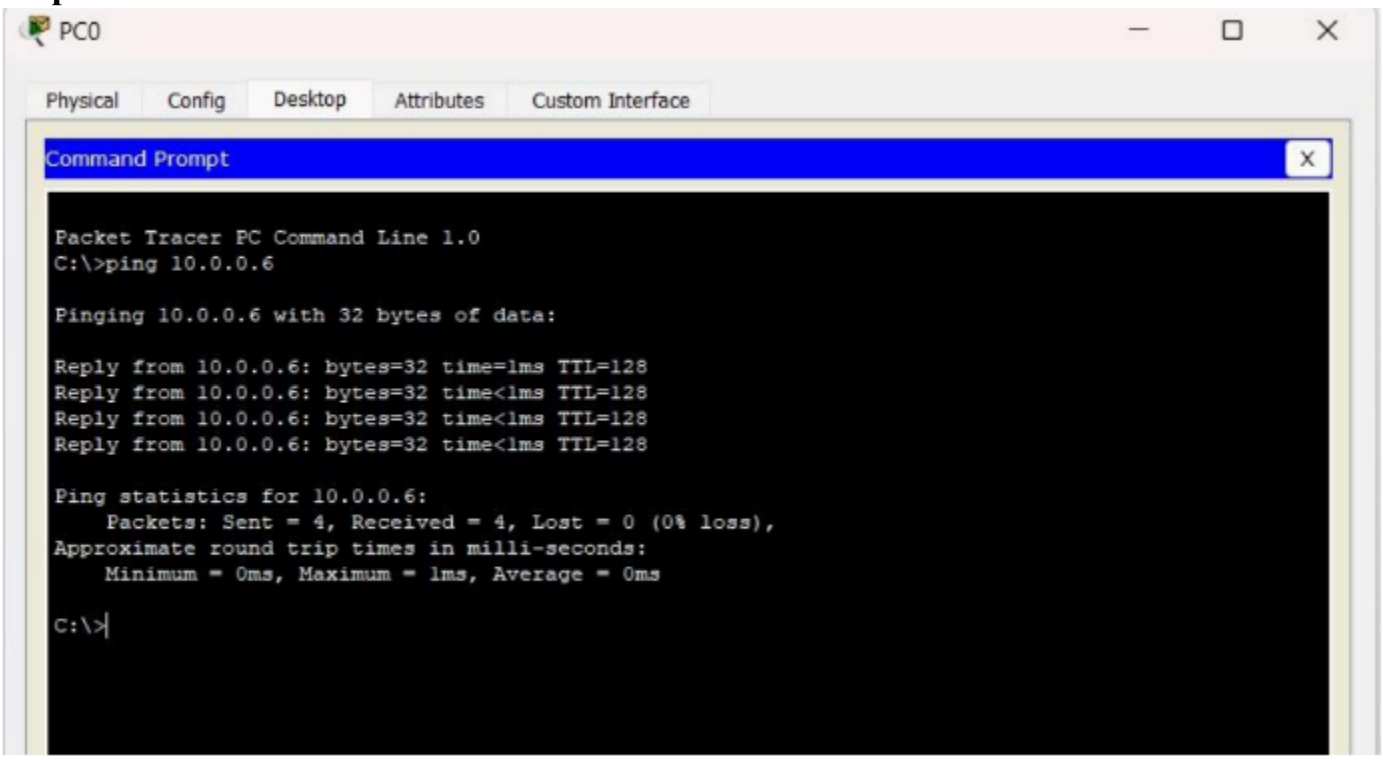
Procedure




PROCEDURE:
1) Add a router, switch, server
2) add 3PC's and they along with server need to be connected to switch through a copper straight, switch is connected to router through copper straight.
3) Click on server → configure gateway 10.0.0.10 and fastethernet 0/0 → IP address 10.0.0.1.
Click on desktop → IP configuration → make it dynamic address from server.
This makes PCs to request dynamic


4) Click on each PC and do the same.
As we change from static to dynamic, we get message "DHCP request successful"

Output

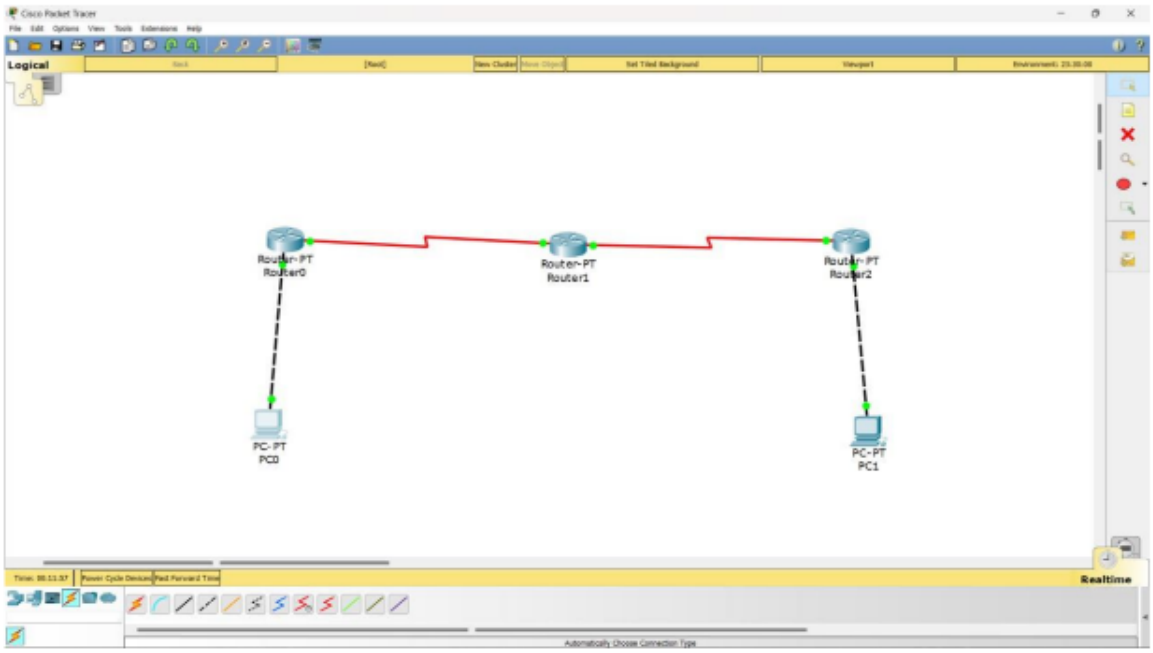

PC0
Physical
Config
Desktop
Attributes
Custom Interface
Command Prompt
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.6
Pinging 10.0.0.6 with 32 bytes of data:
Reply from 10.0.0.6: bytes=32 time=1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128
Reply from 10.0.0.6: bytes=32 time<1ms TTL=128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

PROCEDURE:

1) Set up the 3 routers: router 0, router 1, router 2 & 2 PC's. PC0 and PC1. Connect routers with serial DCE and connect PC's to routers with copper straight
2) Set IP address for each of the PC's as follows:
   PC0: 10.0.0.1
   PC1: 40.0.0.1
   Set gateways of the router as follows:
   Router 0: 10.0.0.10 and 20.0.0.10
   Router 1: 20.0.0.20 and 30.0.0.10
   Router 2: 30.0.0.20 and 40.0.0.10
3) Setting up the PC and router connections in CLI

OBSERVATIONS:

Router 0:

```
Router >enable
Router # config t
Router(config)# interface fa 0/0
Router(config-if)# ip address 10.0.0.10 255.0.0.0
Router(config-if)# no shut
Router(config-if)# exit
Router(config)# interface serial 2/0
Router(config-if)# ip address 20.0.0.10 255.0.0.0
                  # encapsulation ppp
                  # clock rate 64000
                  # no shut
                  # router rip
Router(config-router)# network 10.0.0.0
                     # network 20.0.0.0
                     # exit
Router(config)# exit
Router # show ip route
C  10.0.0.0/8 is directly connected FastEthernet 0/0
```

Router 1:

```
Router > enable
Router # config t
Router(config)# int se2/0
Router(config-if)# ip address 20.0.0.0 255.0.0.0
                 # encapsulation ppp
                 # no shut
                 # exit
Router(config)# int se 3/0
Router(config-if)# ip address 30.0.0.0 255.0.0.0
                 # encapsulation ppp
                 # clock rate 64000
                 # no shut
                 # exit
Router(config)# router rip
Router(config-router)# network 20.0.0.0
                     # network 30.0.0.0
                     # exit
Router(config)# exit
Router # show ip route
R  10.0.0.0/8 [120/1] via 20.0.0.10 00:00:10, serial 2/0
   20.0.0.0/8 is variably subnetted, 2 subnets, 2 mask
C  20.0.0.0/8 is directly connected, serial 2/0
C  20.0.0.0/32 is directly connected, serial 2/0
   30.0.0.0/8 is variably subnetted, 2 subnets, 2 mask
C  30.0.0.0/8 is directly connected, serial 3/0
C  30.0.0.20/32 is directly connected, serial 3/0
```

Router 2:

```
Router > enable
Router # config t
Router(config)# int se2/0
Router(config-if)# ip address 30.0.0.20 255.0.0.0
                 # encapsulation PPP
                 # no shut
                 # exit
Router(config)# int fa 0/0
Router(config-if)# ip address 40.0.0.10 255.0.0.0
                 # no shut
                 # exit
Router(config)# router rip
Router(config-router)# network 30.0.0.0
                     # network 40.0.0.0
                     # exit
Router(config)# exit
Router # show ip route
R  10.0.0.0/8 [120/2] via 30.0.0.10, 00:00:09, serial 2/0
R  20.0.0.0/8 [120/1] via 30.0.0.10, 00:00:09, serial 2/0
   30.0.0.0/8 is variably subnetted, 2 subnets, 2 mask
C  30.0.0.0/8 is directly connected, serial 2/0
C  30.0.0.10/32 is directly connected, serial 2/0
C  40.0.0.0/8 is directly connected, fastethernet 0/0
```

PROCEDURE:

5) The rip command established dynamic connection by setting up rip protocol and is used at clocked connections.
6) After all connections are done and rip is established, ping PC1 from PC0 using CLI.

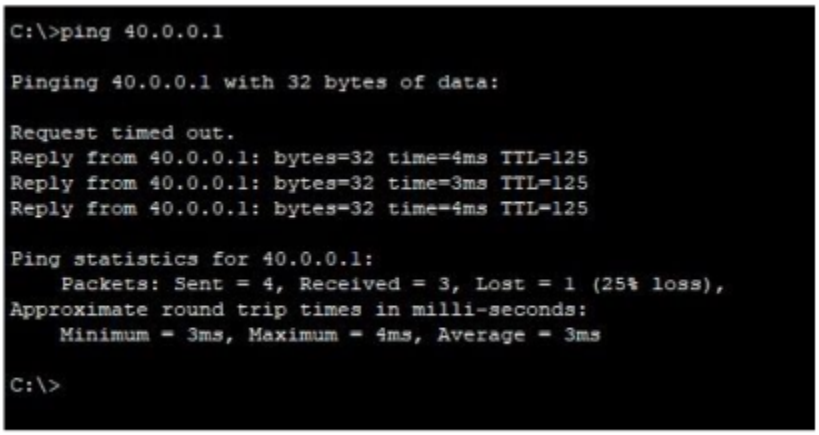
OBSERVATIONS:

```
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=12 ms TTL=125
Reply from 40.0.0.1 : bytes=32 time=6 ms TTL=125
Reply from 40.0.0.1 : bytes=32 time=14 ms TTL=125
Ping statistics for 40.0.0.1 :
    Packets : Sent = 4, Received = 3, Lost = 1 (25% loss)

PC > ping 40.0.0.01
Pinging 40.0.0.01 with 32 bytes of data
Reply from 40.0.0.1 : bytes = 32 time=17ms TTL=125
Reply from 40.0.0.1 : bytes = 32 time =10ms TTL=125
Reply from 40.0.0.1 : bytes = 32 time = 11ms TTL=125
Reply from 40.0.0.1 : bytes = 32 time = 7ms TTL=125
Ping statistics for 40.0.0.1 :
    Packets : Sent = 4, Received = 4, Lost = 0 (0% loss)
```

LEARNINGS:

- Routing Information Protocol (RIP) is a dynamic routing protocol that uses hop count as routing metric to find best path between source and destination network.
- It has two versions:
  - -> RIP v1 - a.k.a classful routing protocol because it does not send information of subnet mask.
  - -> RIP v2 - classless routing protocol, sends information of subnet mask.
- Clock rate is used in DCE serial configuration to configure clock speed of link.

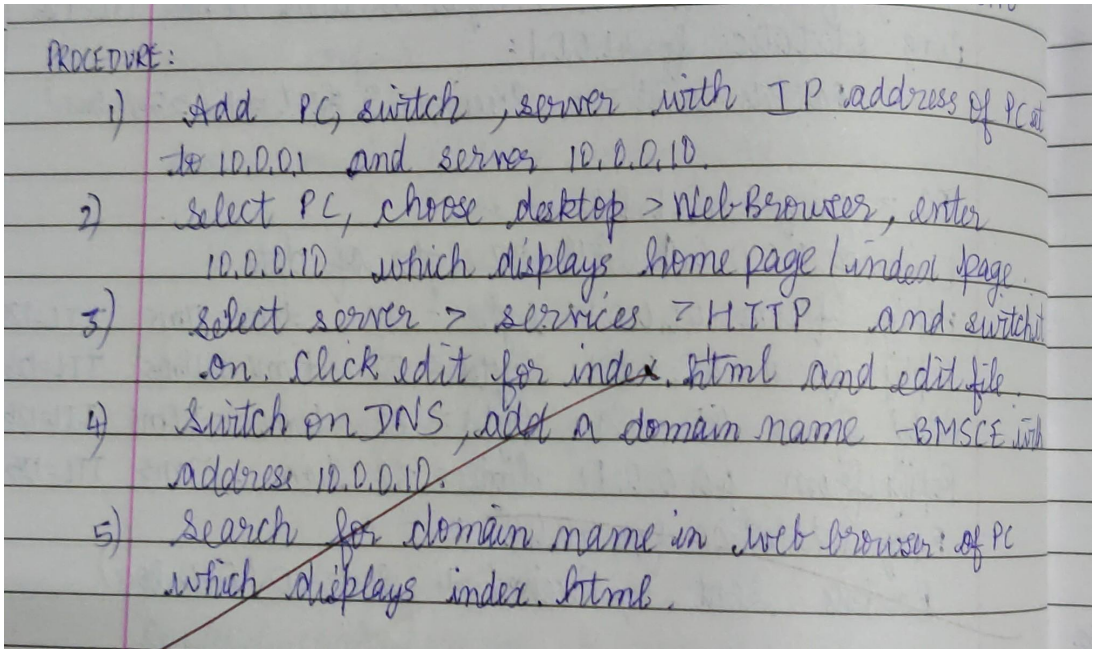
Output:


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 - 6

Aim of the program

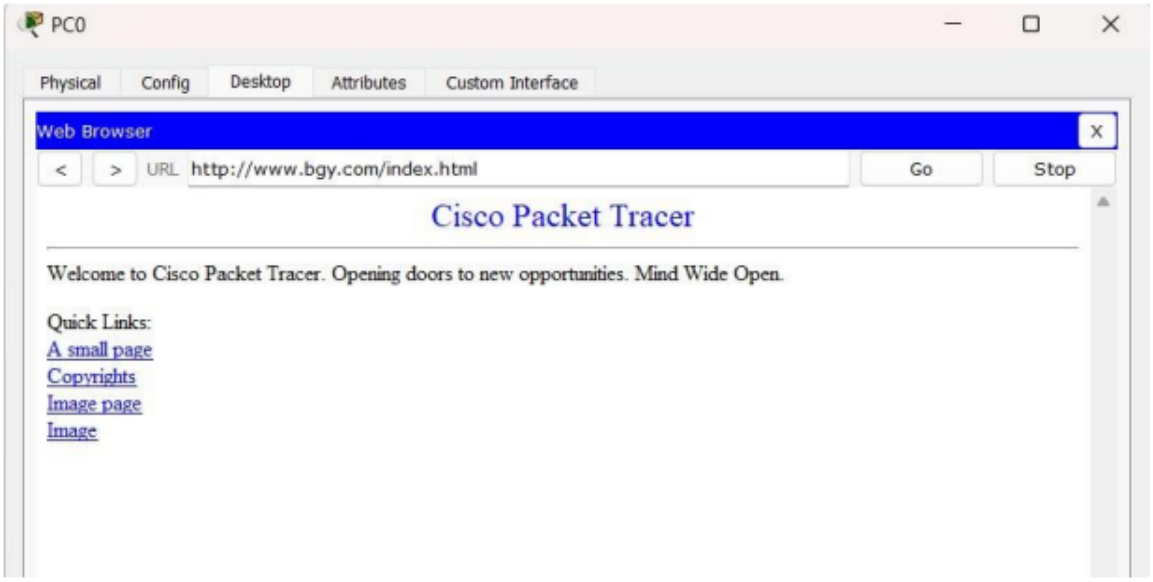
Demonstration of WEB server and DNS using Packet Tracer



PROCEDURE:

1) Add PC, switch, server with IP address of PC as 10.0.0.1 and server 10.0.0.10.
2) Select PC, choose desktop > Web Browser, enter 10.0.0.10 which displays home page / index page.
3) Select server > services > HTTP and switch on. Click edit for index.html and edit file.
4) Switch on DNS, add a domain name - BMSCE with address 10.0.0.10.
5) Search for domain name in web browser of PC which displays index.html.

Output



Cycle-2

Experiment - 1

Aim of the Experiment

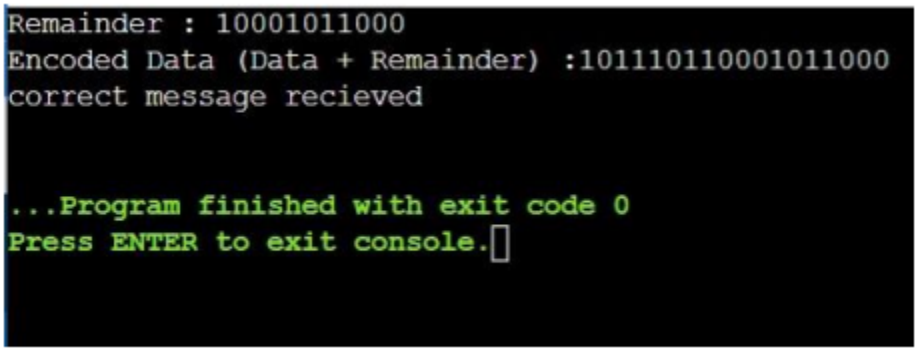
Write a program for error detecting code using CRC-CCITT (16-bits).

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:10001000000100001");

                 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="00000000000000000000";
                 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:



Experiment - 2

Aim of the Experiment

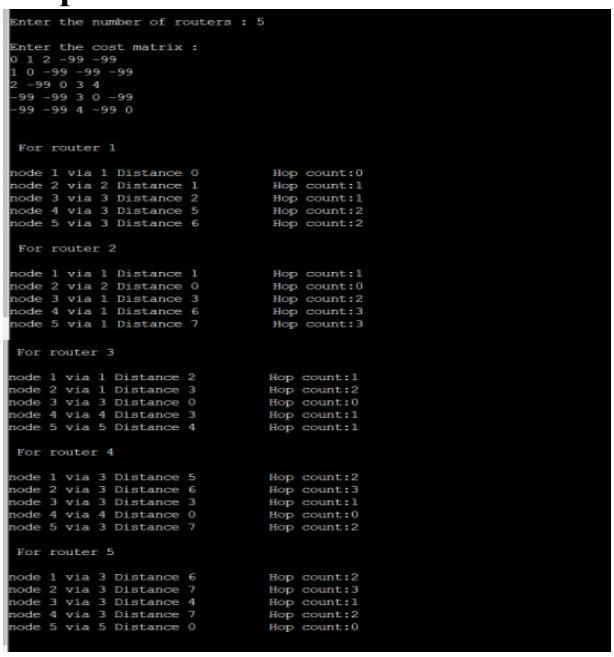
Write a program for distance vector algorithm to find suitable path for transmission.

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 arbitary 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();
}
```

Output:



Experiment - 3

Aim of the Experiment

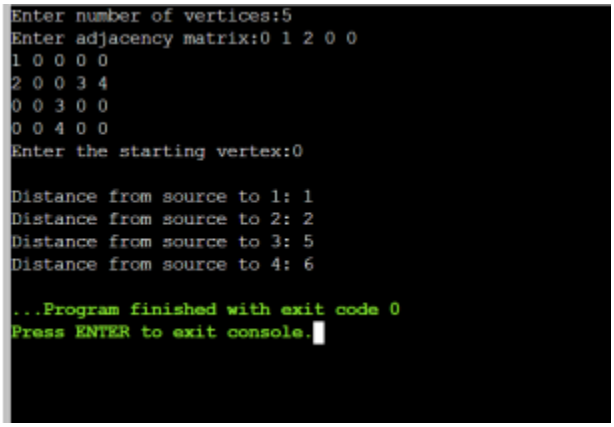
Implement Dijkstra's algorithm to compute the shortest path for a given topology.

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;
}
```

Output:



Experiment - 4

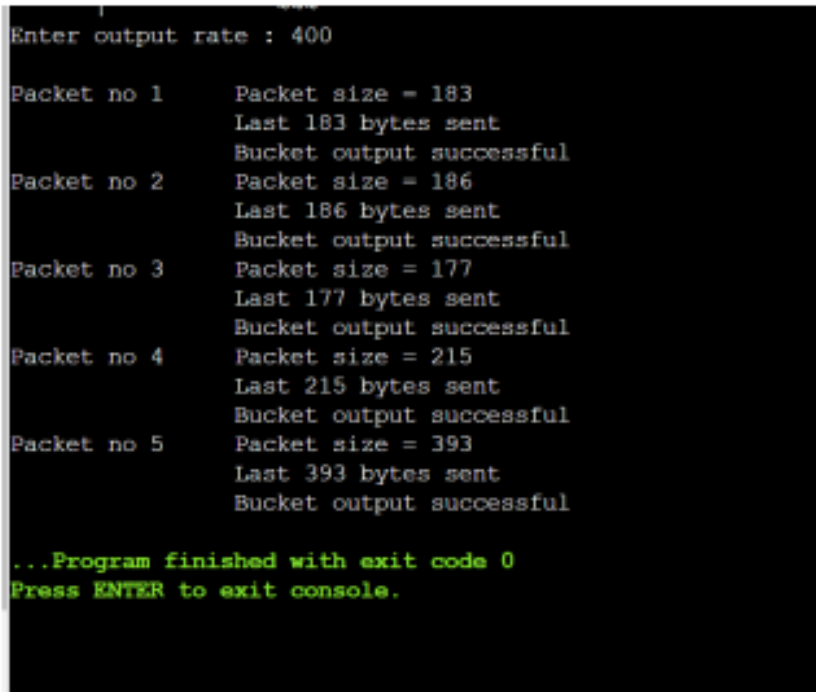
Aim of the Experiment

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 - 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()

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 - 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 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}'
```