VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COURSE TITLE

Submitted by

TARANNUM(1BM20CS171)

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

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
October-2022 to Feb-2023

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**COMPUTER NETWORKS "** is carried out by **TARANNUM S (1BM20CS171),** who is a 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 G S
Name of the Lab-Incharge

Designation

Department of CSE

BMSCE, Bengaluru

Dr. Jyothi S Nayak

Professor and Head

Department of CSE

BMSCE, Bengaluru

`

Index

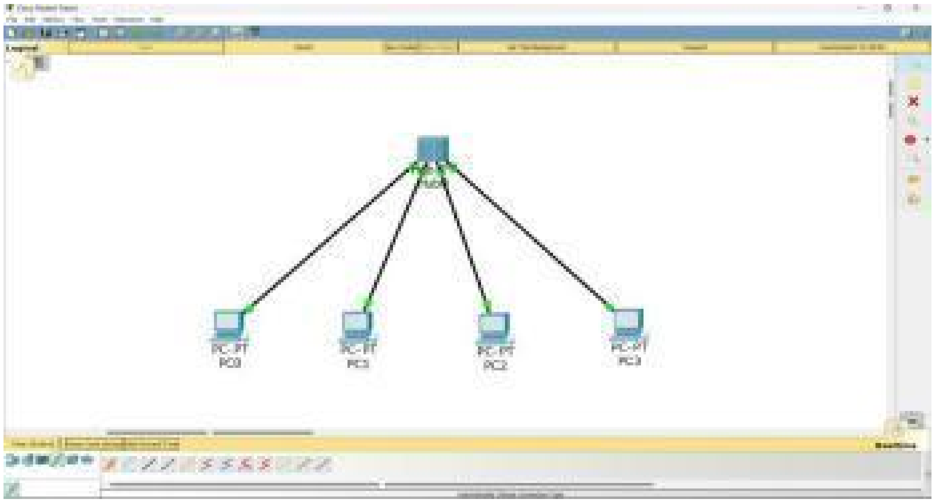
Cycle-1

Experiment No 1

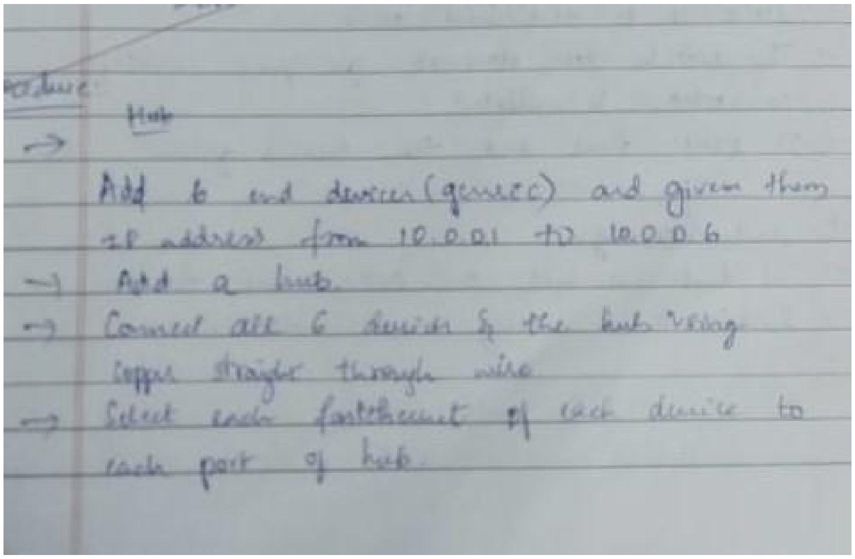
Aim of the program

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

Hub Topology



Procedure:



cedure

Hub

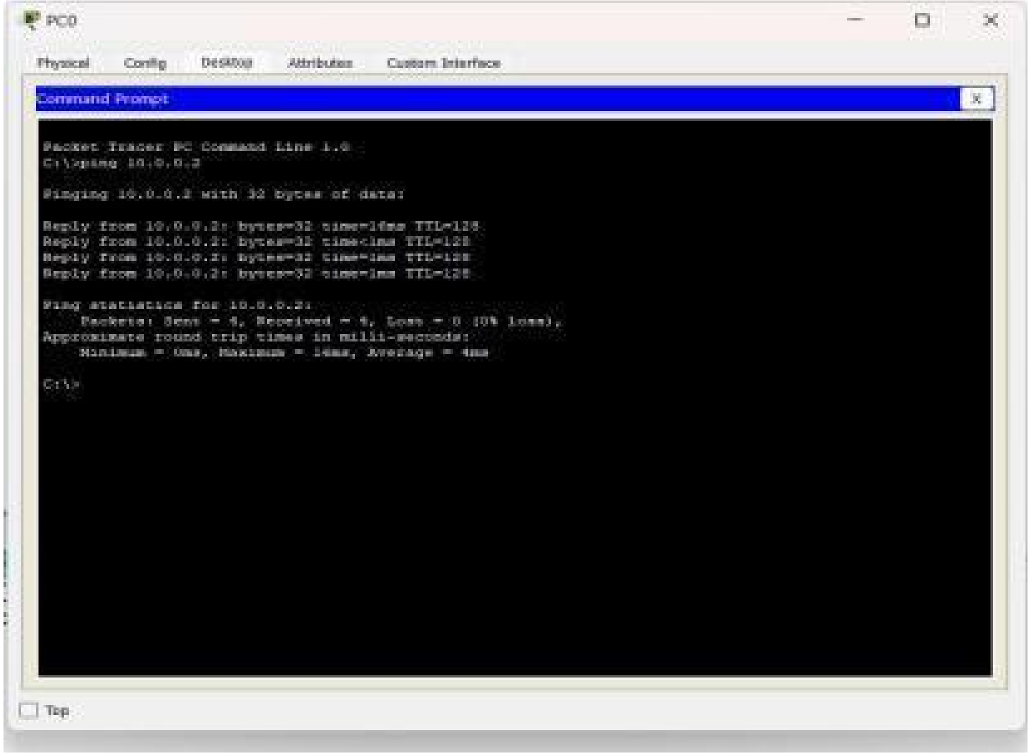
→ Add 6 end devices(generic) and given them IP address from 10.0.0.1 to 10.0.0.6

→ Add a hub.

→ Connect all 6 devices & the hub using copper straight through wire

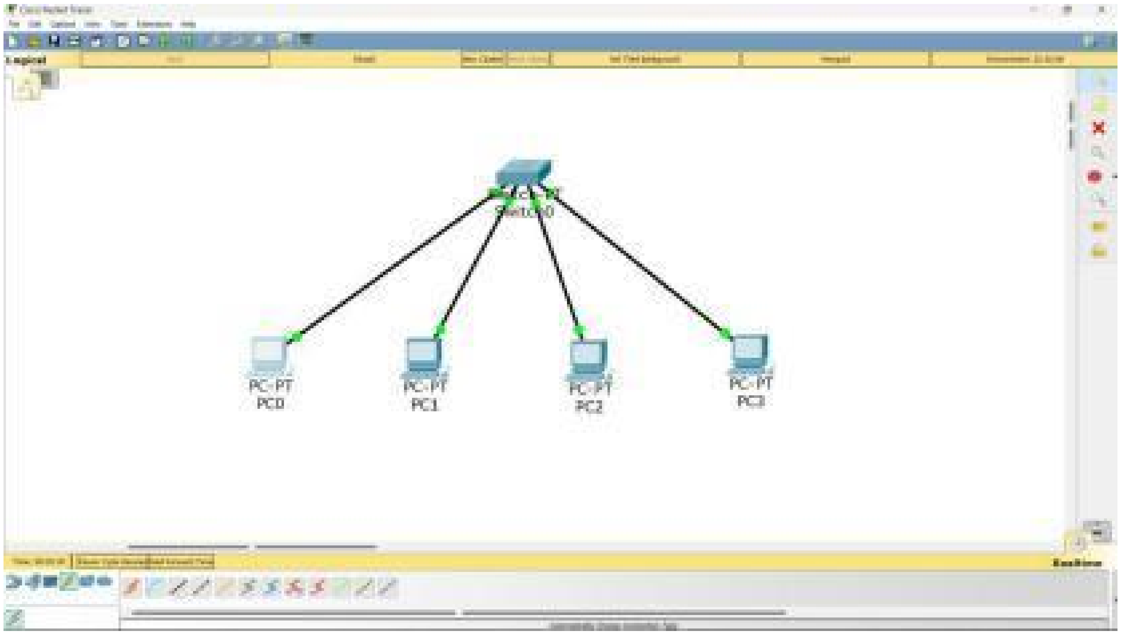
→ Select each fastethernet of each device to each port of hub.

Output:

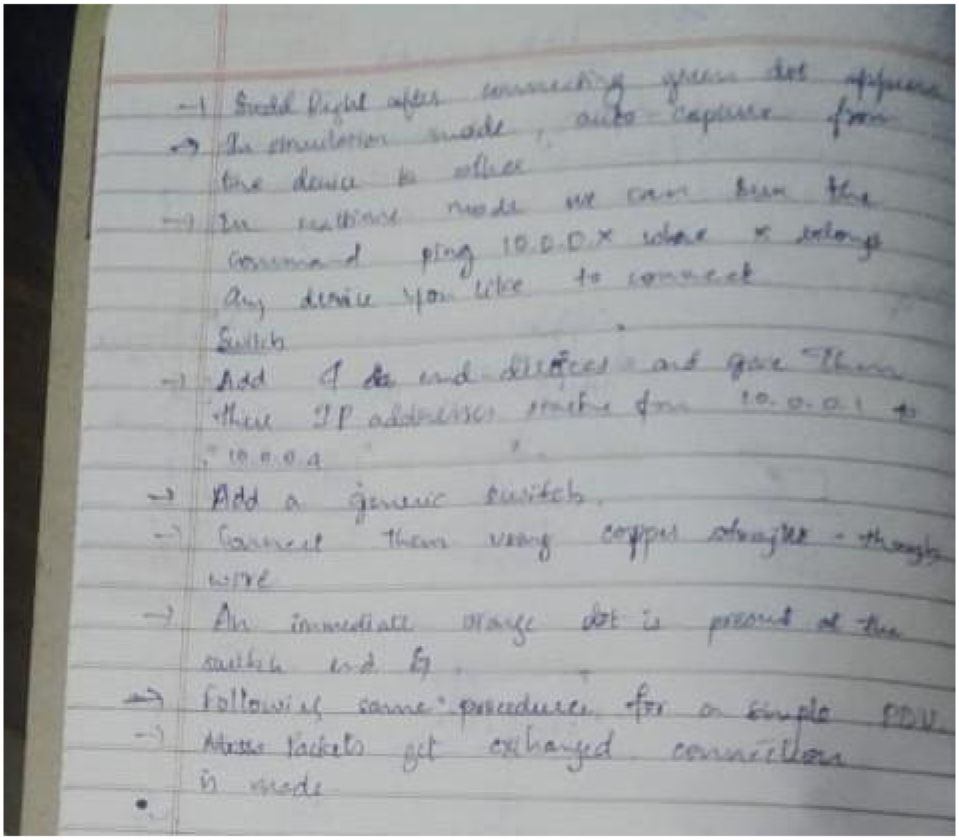


```
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

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

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

C:\>
```

Switch Topology:



Procedure:

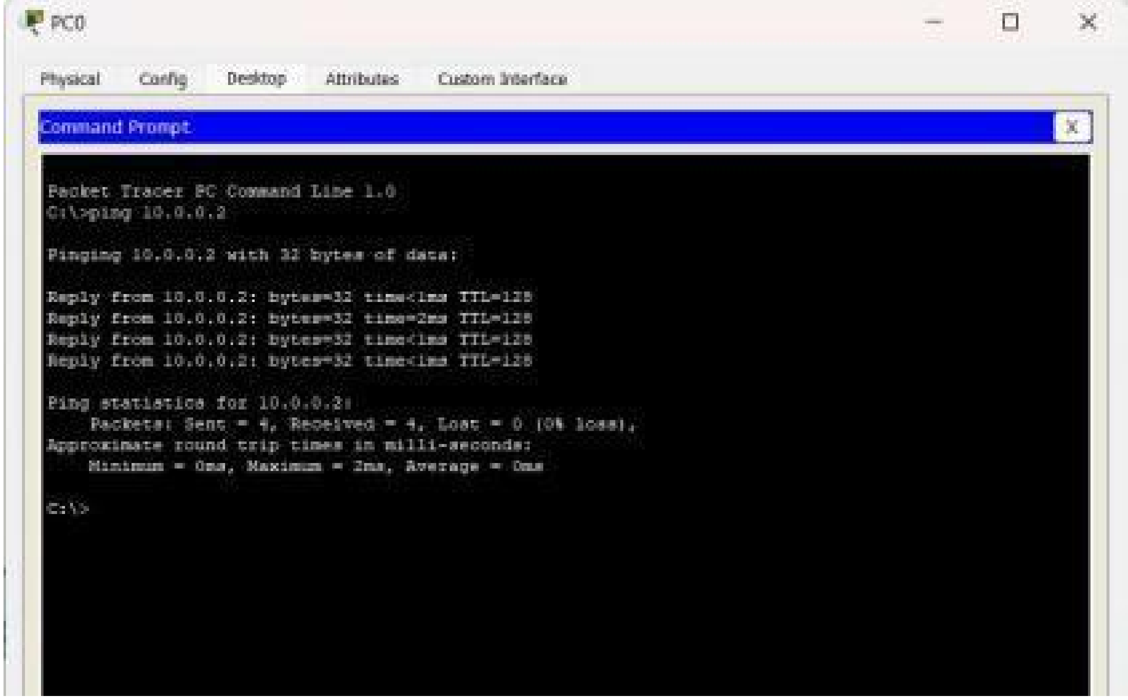


→ Solid light after connecting green dot appears
→ In simulation mode, auto capture from the device to other
→ In realtime mode we can see the command ping 10.0.0.x where x belongs any device you like to connect

Switch

→ Add 4 ~~de~~ end devices and give them their IP addresses starting from 10.0.0.1 to 10.0.0.4
→ Add a generic switch.
→ Connect them using copper straight-through wire
→ An immediate orange dot is present at the switch end by.
→ Following same procedure for a simple PDU
→ these packets get exchanged connections is made

Output:



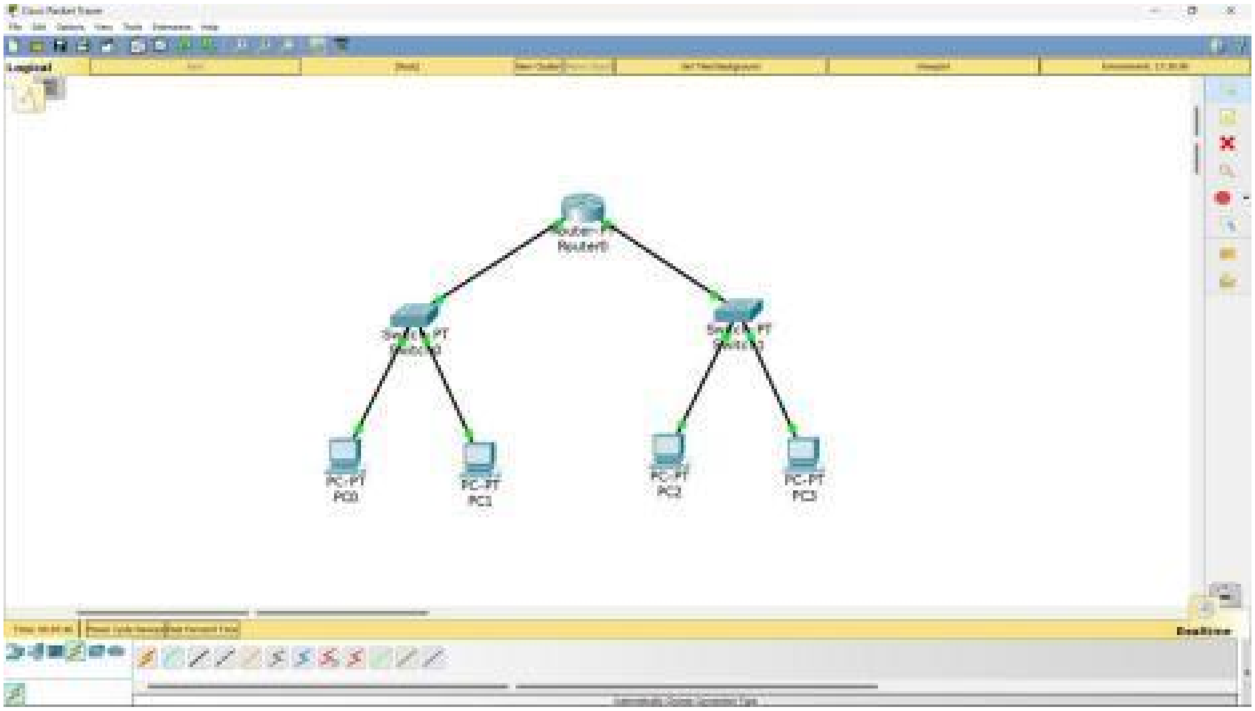
```
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.2

Pinging 10.0.0.2 with 32 bytes of data:

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

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

C:\>
```

Experiment No 2

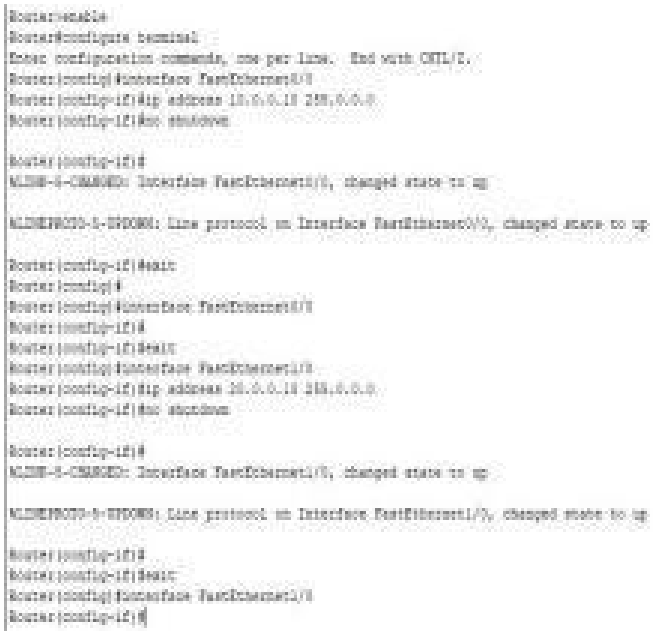
Aim of the program

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

Topology

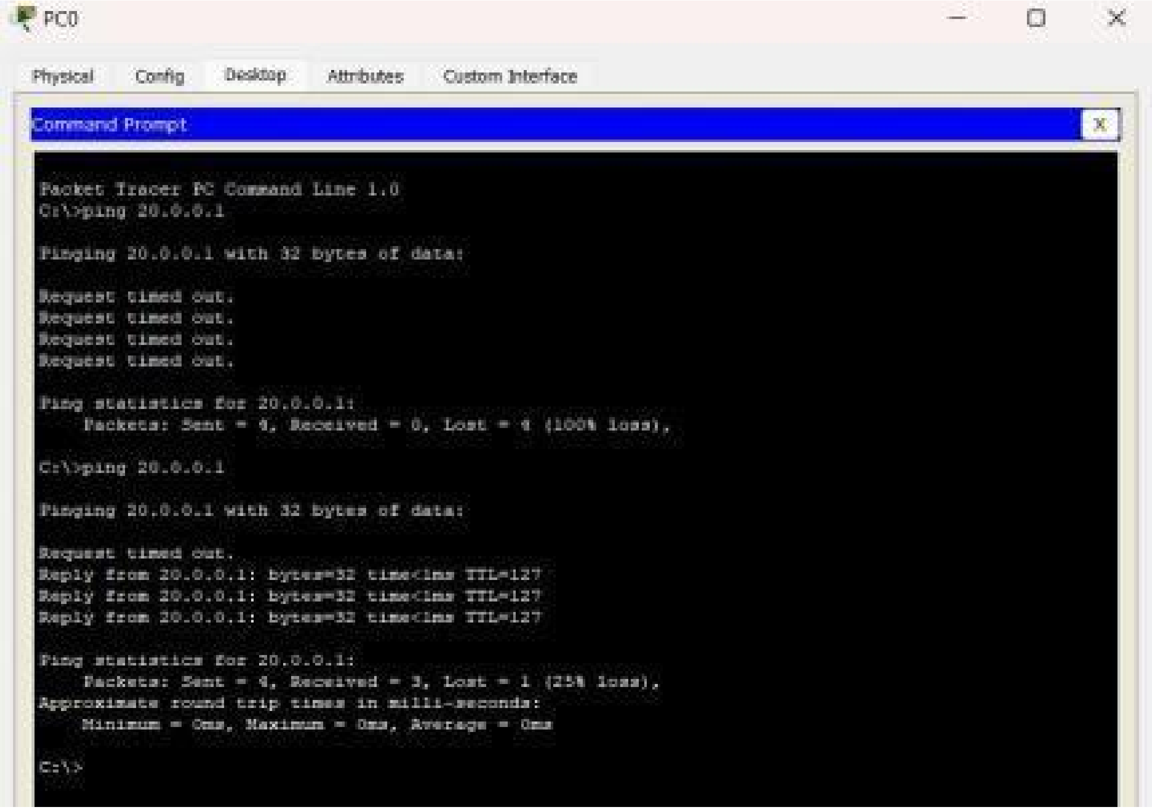


Procedure:



```
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)#
```

Output:



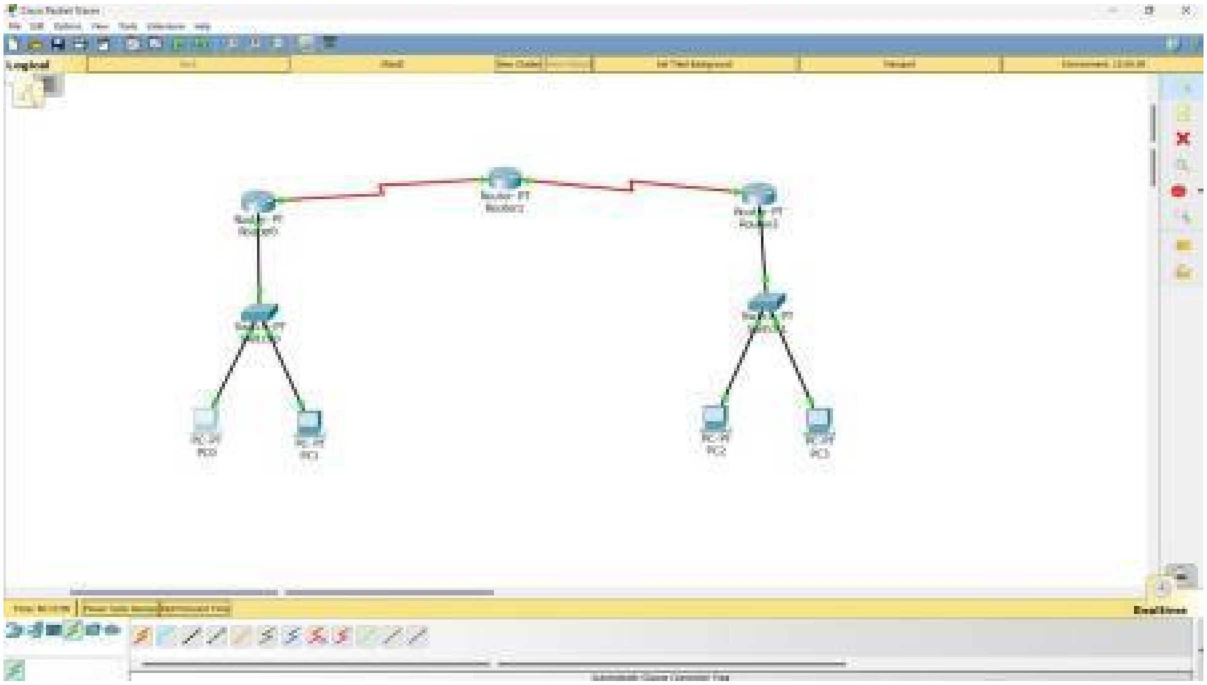
```
Packet Tracer PC Command Line 1.0
C:\>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

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

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

C:\>ping 20.0.0.1

Pinging 20.0.0.1 with 32 bytes of data:

Request timed out.
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127
Reply from 20.0.0.1: bytes=32 time<1ms TTL=127

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

C:\>
```

Experiment No 3

Aim of the program

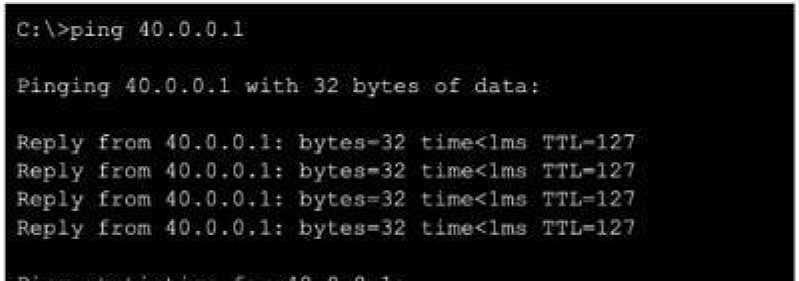
Configuring static and default route to the Router

Topology for static routing



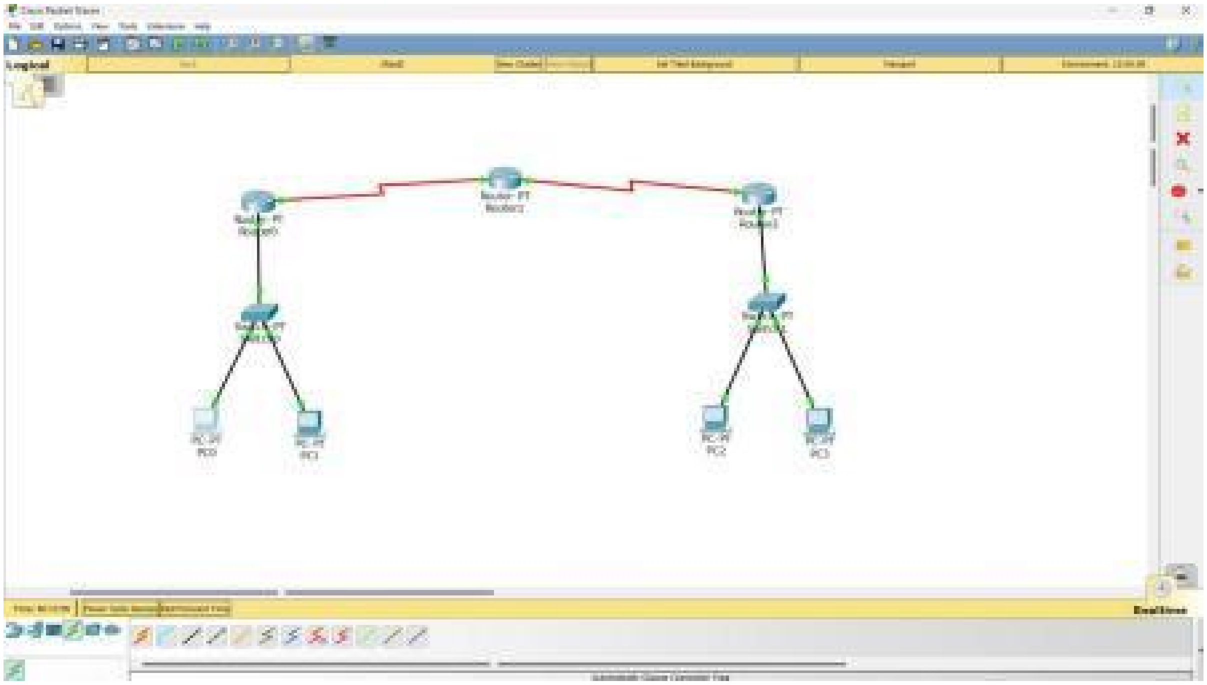
Procedure:

Output:



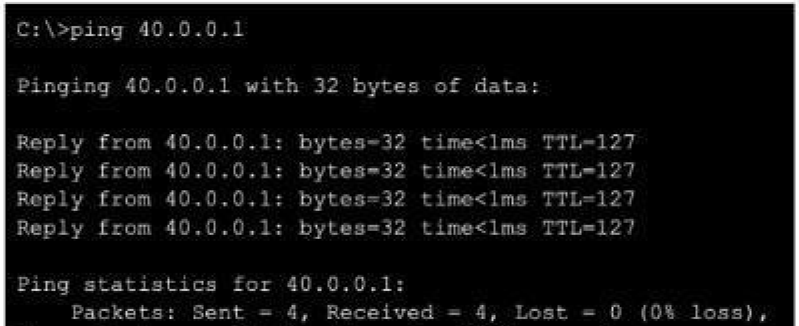
```
C:\>ping 40.0.0.1

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
```

Topology for default routing



Procedure

Output:



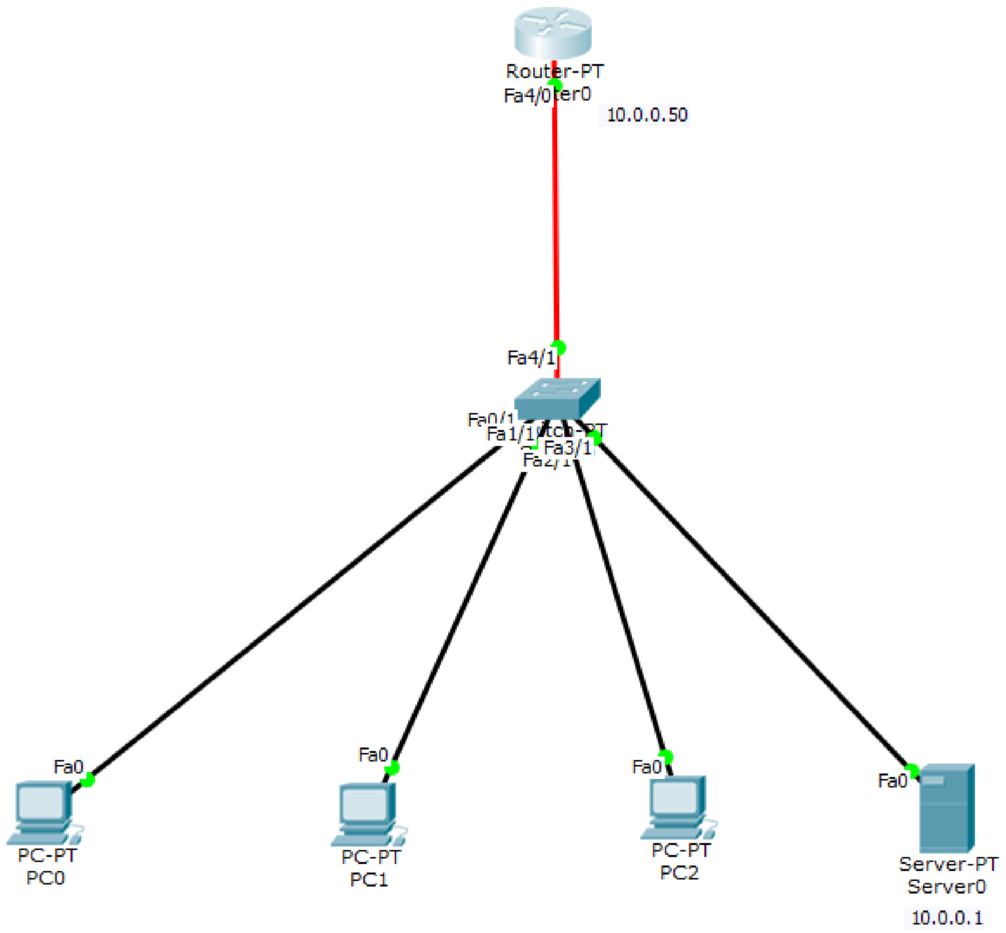
```
C:\>ping 40.0.0.1

Pinging 40.0.0.1 with 32 bytes of data:

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

Experiment No 4

Aim of the program

Configuring DHCP within a LAN in a packet Tracer

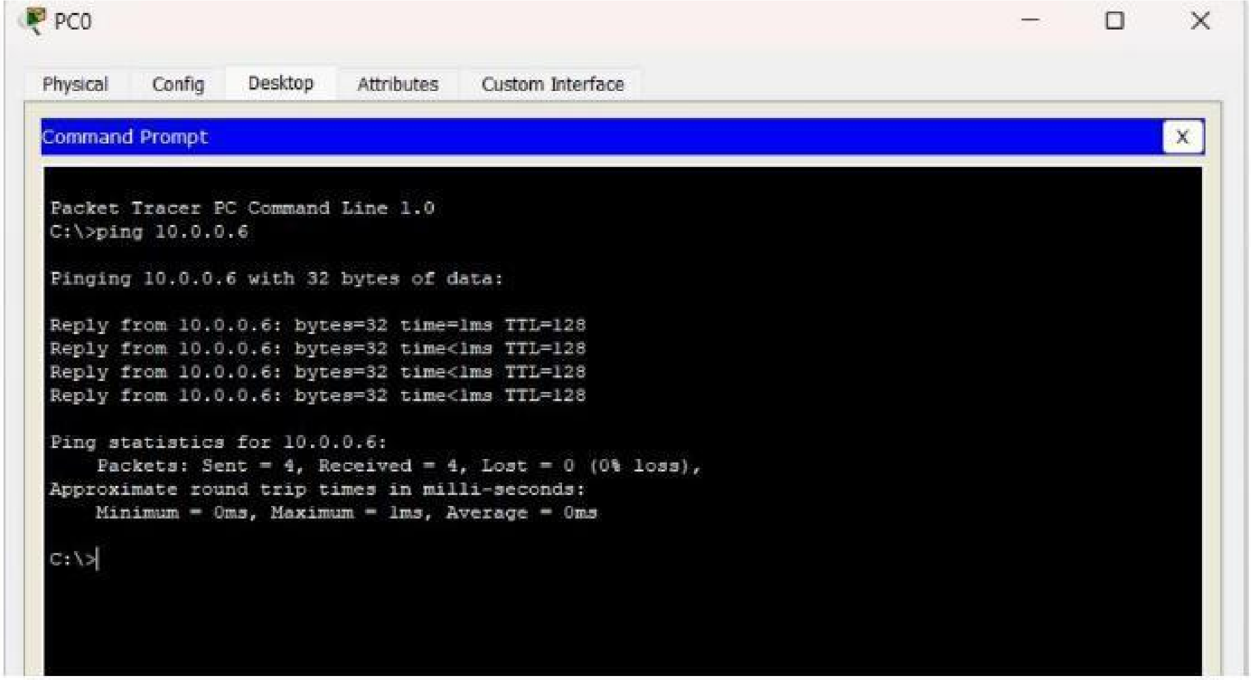
Topology


Router-PT
Fa4/0
Router0
10.0.0.50
Fa4/1
Fa0/1
Fa1/1
Switch-PT
Fa3/1
Fa2/1
Fa0
PC-PT
PC0
Fa0
PC-PT
PC1
Fa0
PC-PT
PC2
Fa0
Server-PT
Server0
10.0.0.1

Procedure:

```
          --- System Configuration Dialog ---

Continue with configuration dialog? [yes/no]: no


Press RETURN to get started!



Router>enable
Router#config t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#interface fastethernet4/0
Router(config-if)#ip address 10.0.0.50 255.0.0.0
Router(config-if)#no shut

Router(config-if)#
%LINK-5-CHANGED: Interface FastEthernet4/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface FastEthernet4/0, changed state to up
exit
Router(config)#
```

Output:



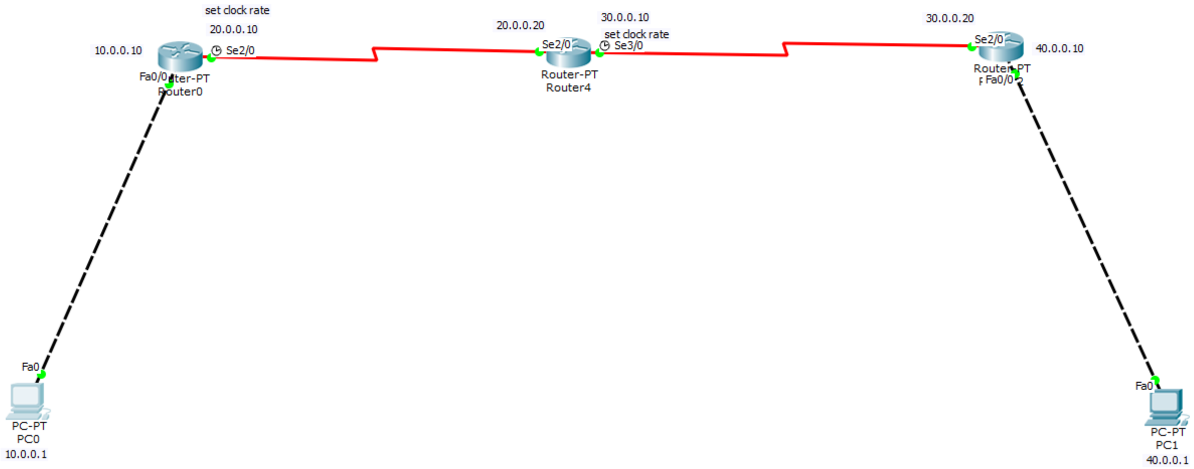
```
Packet Tracer PC Command Line 1.0
C:\>ping 10.0.0.6

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

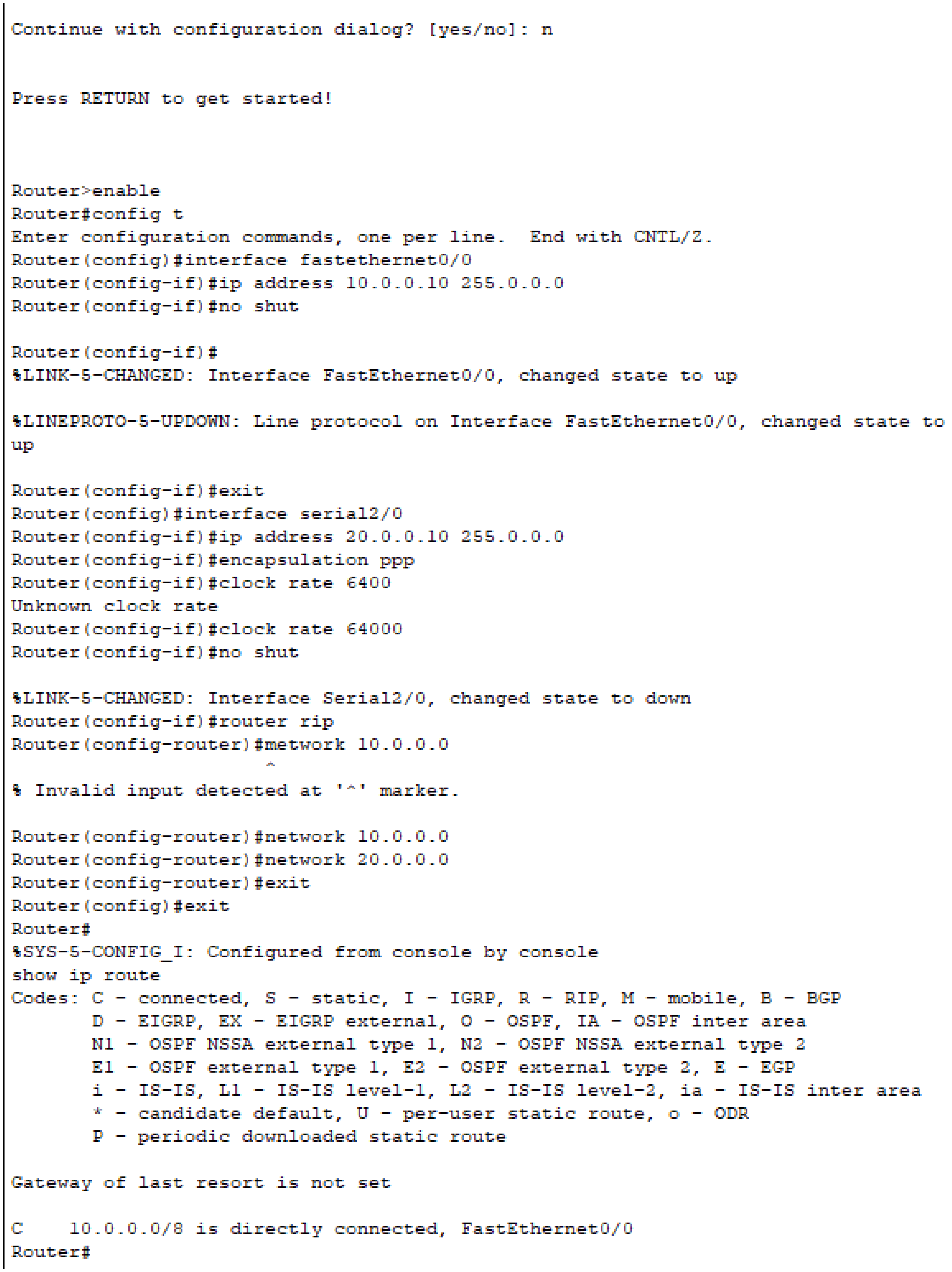
Aim of the program

Configuring RIP Routing Protocol in Routers

Topology:

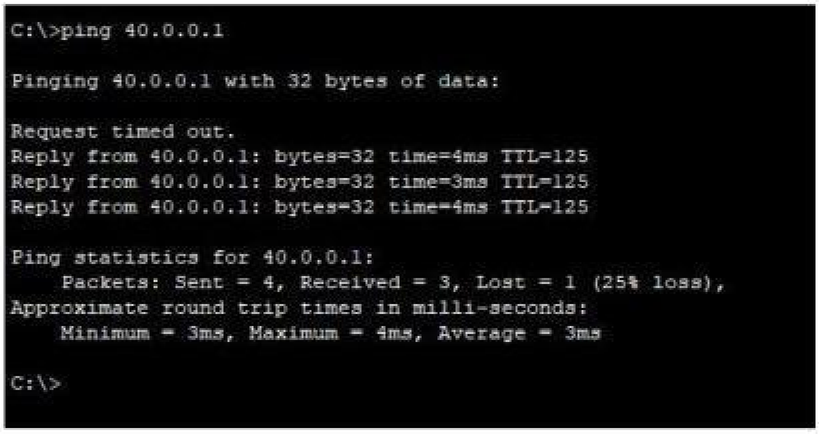


Procedure:



```
Continue with configuration dialog? [yes/no]: n


Press RETURN to get started!



Router>enable
Router#config t
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 shut

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)#interface serial2/0
Router(config-if)#ip address 20.0.0.10 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#clock rate 6400
Unknown clock rate
Router(config-if)#clock rate 64000
Router(config-if)#no shut

%LINK-5-CHANGED: Interface Serial2/0, changed state to down
Router(config-if)#router rip
Router(config-router)#metwork 10.0.0.0
                      ^
% Invalid input detected at '^' marker.

Router(config-router)#network 10.0.0.0
Router(config-router)#network 20.0.0.0
Router(config-router)#exit
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
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
Router#
```

Output:



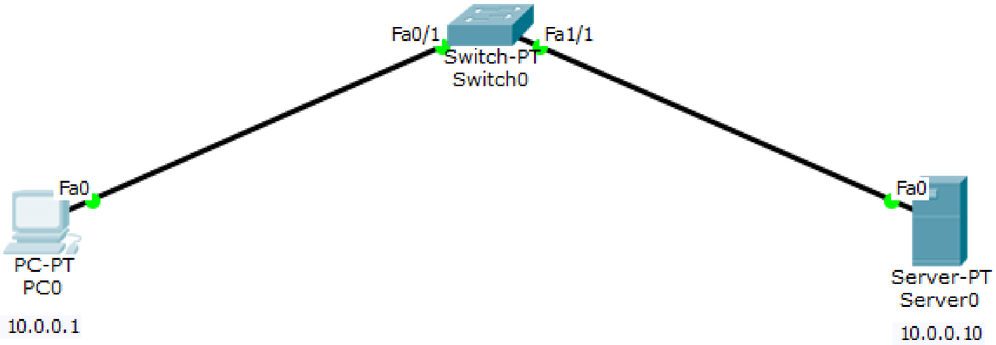
```
C:\>ping 40.0.0.1

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

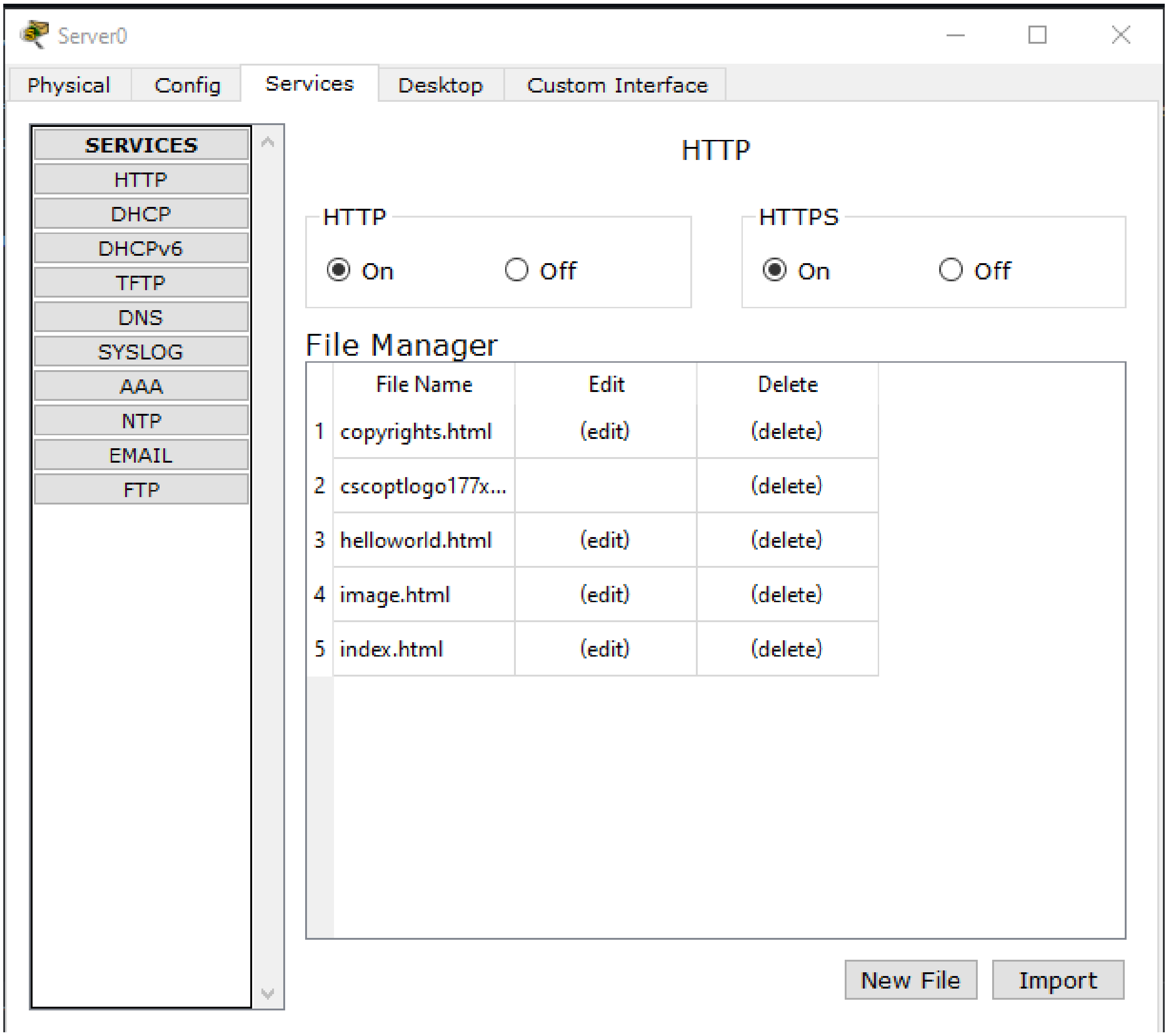
Aim of the program

Demonstration of WEB server and DNS using Packet Tracer

Topology:



Procedure:



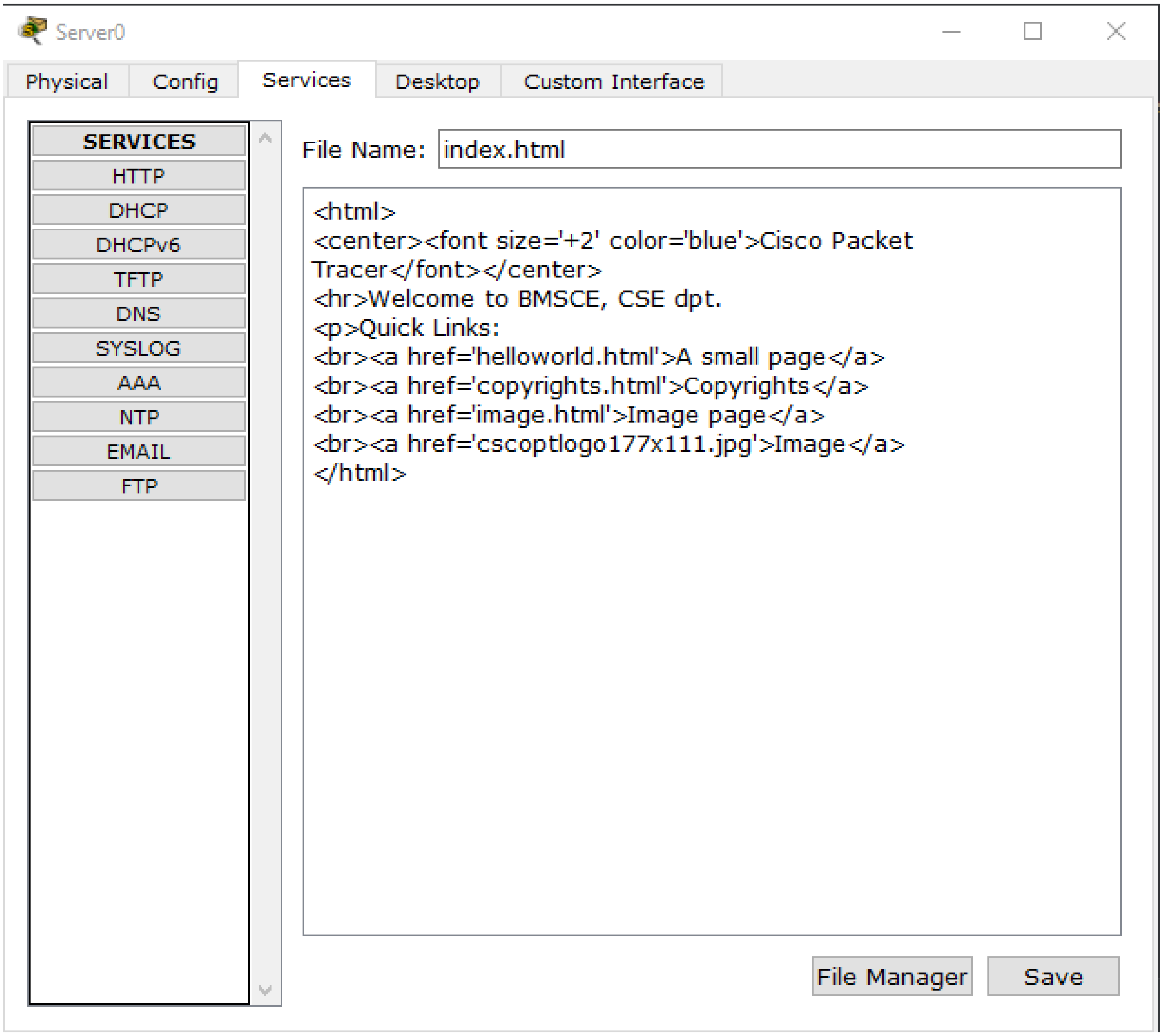
HTTP

HTTP ● On ○ Off

HTTPS ● On ○ Off

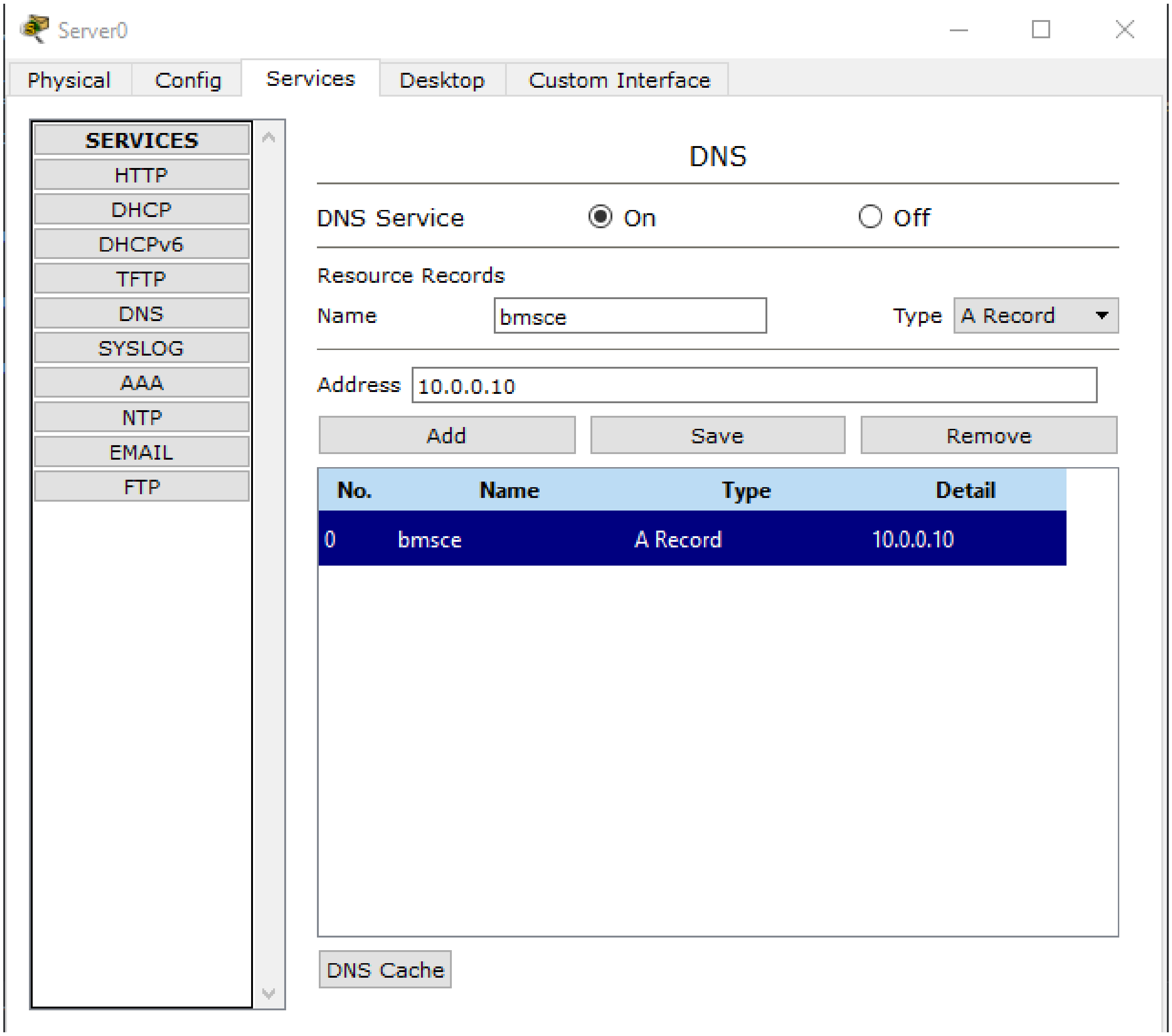
File Manager

	File Name	Edit	Delete
1	copyrights.html	(edit)	(delete)
2	cscoptlogo177x...		(delete)
3	helloworld.html	(edit)	(delete)
4	image.html	(edit)	(delete)
5	index.html	(edit)	(delete)



File Name: index.html

```
<html>
<center><font size='+2' color='blue'>Cisco Packet
Tracer</font></center>
<hr>Welcome to BMSCE, CSE dpt.
<p>Quick Links:
<br><a href='helloworld.html'>A small page</a>
<br><a href='copyrights.html'>Copyrights</a>
<br><a href='image.html'>Image page</a>
<br><a href='cscoptlogo177x111.jpg'>Image</a>
</html>
```



DNS

DNS Service ● On ○ Off

Resource Records

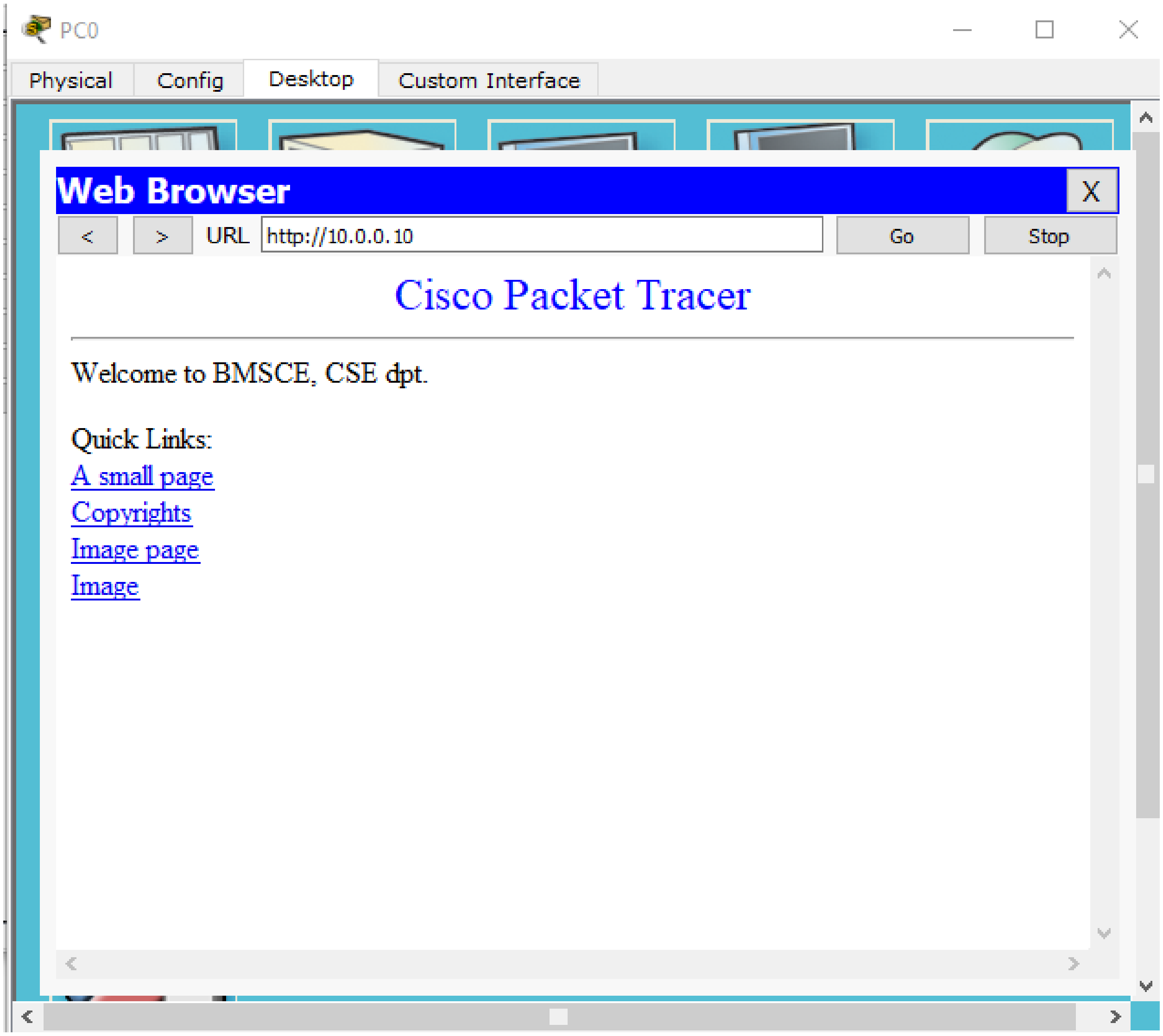
Name: bmsce

Type: A Record

Address: 10.0.0.10

No.	Name	Type	Detail
0	bmsce	A Record	10.0.0.10

Output:



http://10.0.0.10

Cisco Packet Tracer

Welcome to BMSCE, CSE dpt.

Quick Links:
A small page
Copyrights
Image page
Image

Cycle-2

Experiment No 1

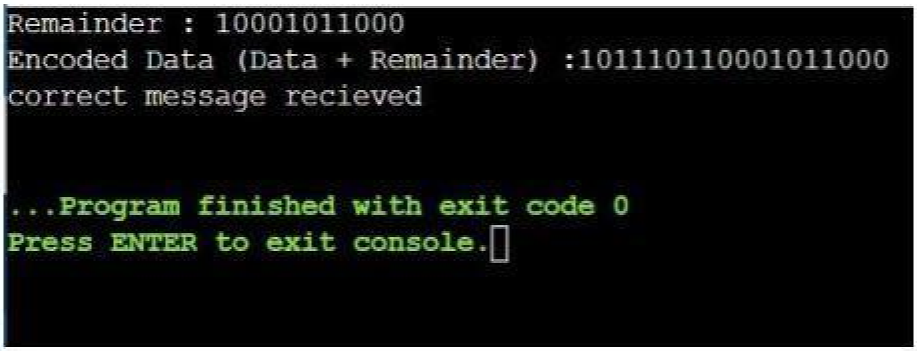
Aim of the Experiment

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

Code:

```
def xor(a, b):

        result = []

        for i in range(1, len(b)):
        if a[i] == b[i]:
            result.append('0')
        else:
            result.append('1')

    return ''.join(result)


def mod2div(dividend, divisor):

        pick = len(divisor)

        tmp = dividend[0 : pick]

    while pick < len(dividend):

        if tmp[0] == '1':

            tmp = xor(divisor, tmp) + dividend[pick]

        else:
            tmp = xor('0'*pick, tmp) + dividend[pick]


        pick += 1

    if tmp[0] == '1':
        tmp = xor(divisor, tmp)
    else:
        tmp = xor('0'*pick, tmp)

    checkword = tmp
    return checkword


def encodeData(data, key):

    l_key = len(key)


    appended_data = data + '0'*(l_key-1)
    remainder = mod2div(appended_data, key)

    codeword = data + remainder
    print("Remainder : ", remainder)
    print("Encoded Data (Data + Remainder) : ",
        codeword)

data = "10001000000100001"
key = "1101"
encodeData(data, key)
```

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

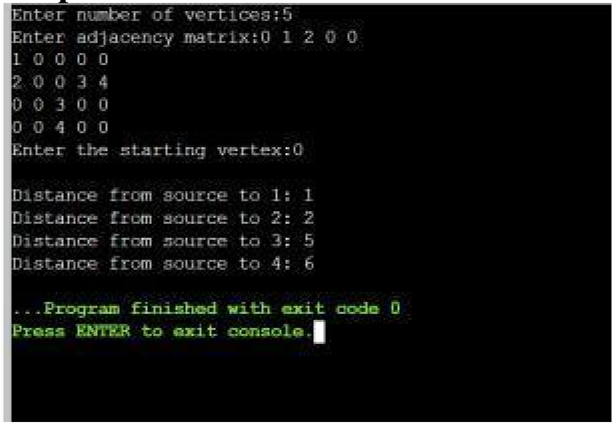
Aim of the Experiment

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

Code

```
class Graph():

    def __init__(self, vertices):
        self.V = vertices
        self.graph = [[0 for column in range(vertices)]
                        for row in range(vertices)]

    def printSolution(self, dist):
        print("Vertex \t Distance from Source")
        for node in range(self.V):
            print(node, "\t\t", dist[node])


    def minDistance(self, dist, sptSet):

        min = 1e7



        for v in range(self.V):
            if dist[v] < min and sptSet[v] == False:
                min = dist[v]
                min_index = v

        return min_index


    def dijkstra(self, src):

        dist = [1e7] * self.V
        dist[src] = 0
        sptSet = [False] * self.V

        for cout in range(self.V):


            u = self.minDistance(dist, sptSet)
            sptSet[u] = True


            for v in range(self.V):
                if (self.graph[u][v] > 0 and
                sptSet[v] == False and
                dist[v] > dist[u] + self.graph[u][v]):
                    dist[v] = dist[u] + self.graph[u][v]

        self.printSolution(dist)

g = Graph(9)
g.graph = [[0, 4, 0, 0, 0, 0, 0, 8, 0],
        [4, 0, 8, 0, 0, 0, 0, 11, 0],
        [0, 8, 0, 7, 0, 4, 0, 0, 2],
        [0, 0, 7, 0, 9, 14, 0, 0, 0],
        [0, 0, 0, 9, 0, 10, 0, 0, 0],
        [0, 0, 4, 14, 10, 0, 2, 0, 0],
        [0, 0, 0, 0, 0, 2, 0, 1, 6],
        [8, 11, 0, 0, 0, 0, 1, 0, 7],
        [0, 0, 2, 0, 0, 0, 6, 7, 0]
        ]

g.dijkstra(0)
```

Output:



```
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 3

Aim of the Experiment

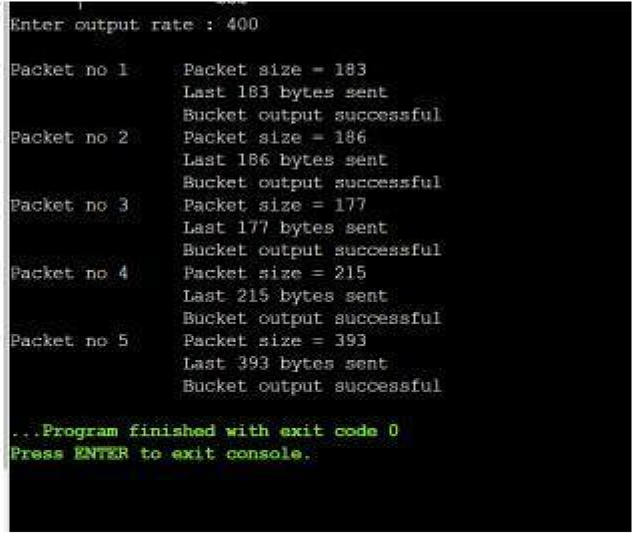
Write a program for congestion control using a leaky bucket algorithm.

CODE

```
inputPacket
=0
                outputPacket=3
                BucketBuffer=0
                time =0



                while(time<10):
                  print("give input packet at time " + str(time))
                  inputPacket= int(input())
                  BucketBuffer=BucketBuffer+inputPacket
                  if BucketBuffer>60:
                     time=time+1
                     BucketBuffer=BucketBuffer-inputPacket
                     print("packet couldnot be stored in bucket")
                     BucketBuffer=BucketBuffer-outputPacket
                     print("Packet of size " + str(outputPacket) + " is removed")
                     print("Bucket has "+ str(BucketBuffer) + " packets")

                  else:
                     time=time+1
                     if BucketBuffer<outputPacket:
```

```
        print("Packet of size " + str(BucketBuffer) + " is removed")
        BucketBuffer=0
        print("Bucket has "+ str(BucketBuffer) + " packets")

    else:
        BucketBuffer=BucketBuffer-outputPacket
        print("Packet of size " + str(outputPacket) + " is removed")
        print("Bucket has "+ str(BucketBuffer) + " packets")
```

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 4

Aim of the Experiment

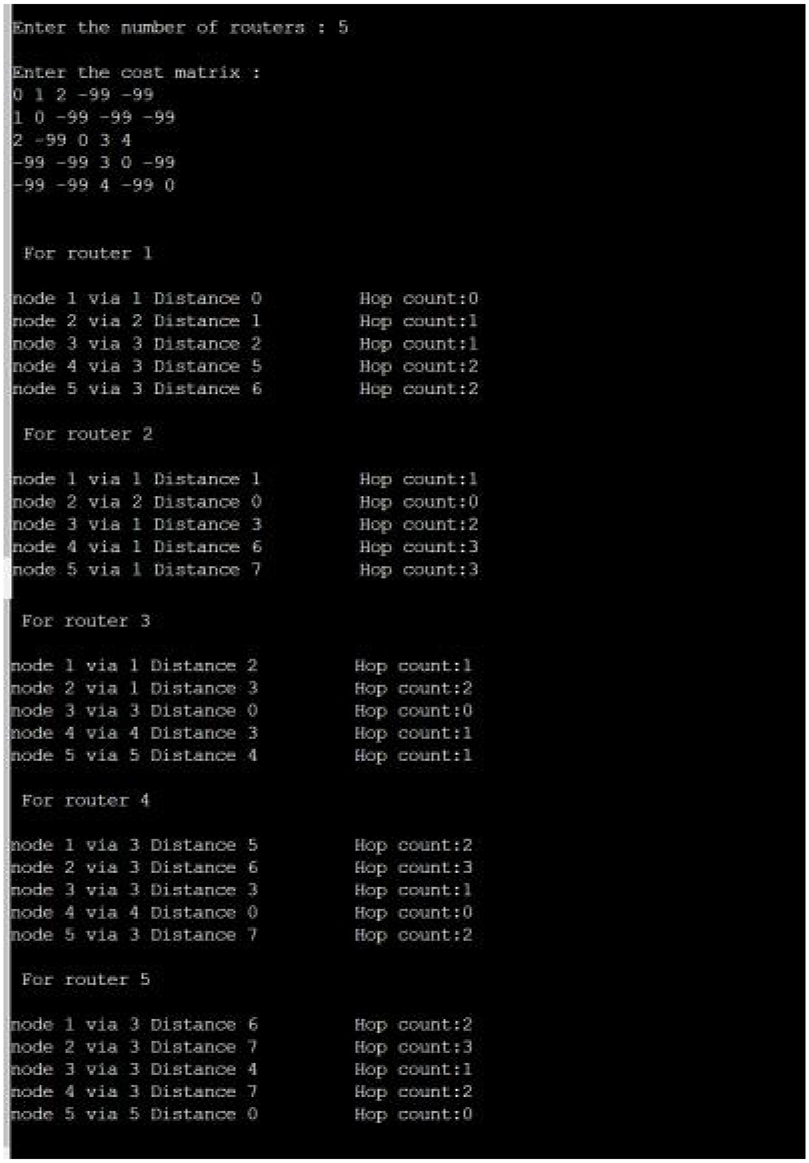
Write a program for a distance vector algorithm to find a suitable path for transmission.

Code

```
#include<stdio.h>
struct node
{
  unsigned dist[20];
  unsigned from[20];
}rt[10];
int main()
{
  int costmat[20][20];
  int nodes,i,j,k,count=0;
  printf("\nEnter the number of nodes : ");
  scanf("%d",&nodes);//Enter the nodes
  printf("\nEnter the cost matrix :\n");
  for(i=0;i<nodes;i++)
  {
     for(j=0;j<nodes;j++)
     {
        scanf("%d",&costmat[i][j]);
        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(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].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("\n\n");
   //getch();
}
```

Output:



```
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 5

Aim of the Experiment

Using TCP/IP sockets, write a client-server program to make the client send the file name and the server to send back the contents of the requested file if present.

Code

Server:

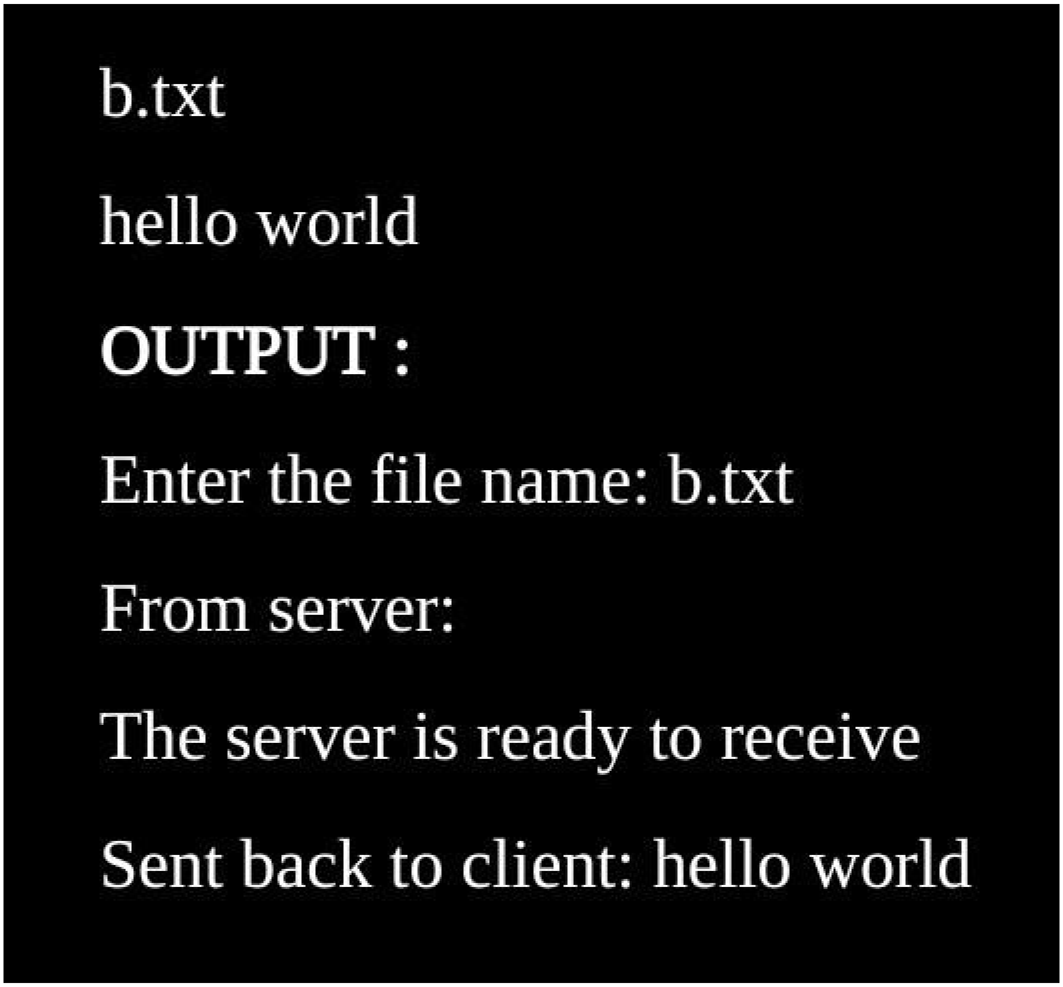
```
from socket import *
serverName = ''
serverPort = 12530
 serverSocket = socket(AF_INET,SOCK_STREAM)

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:



```
b.txt

hello world

OUTPUT :

Enter the file name: b.txt

From server:

The server is ready to receive

Sent back to client: hello world
```

Experiment No 6

Aim of the Experiment

Using UDP sockets, write a client-server program to make the client send the file name and the server to send back the contents of the requested file if present.

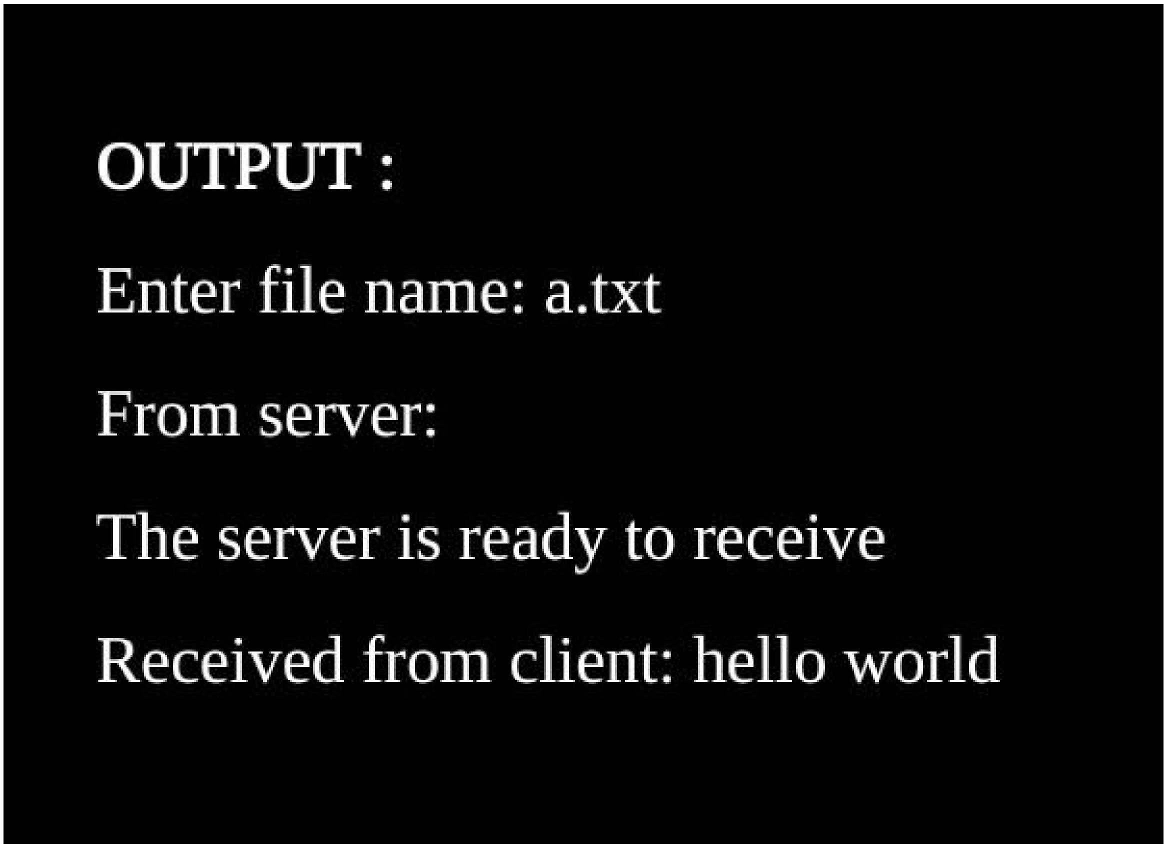
Code

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



OUTPUT :

```
Enter file name: a.txt

From server:

The server is ready to receive

Received from client: hello world
```