VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

COMPUTER NETWORKS

Submitted by

NAYANTARA K KUMAR (1BM20CS228)

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

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

B. M. S. College of Engineering,

Bull Temple Road, Bangalore 560019

(Affiliated To Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "LAB COURSE **COMPUTER NETWORKS**" carried out by **NAYANTARA K KUMAR (1BM20CS228),** 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.

M Lakshmi Neelima
Assistant Professor
Department of CSE
BMSCE, Bengaluru

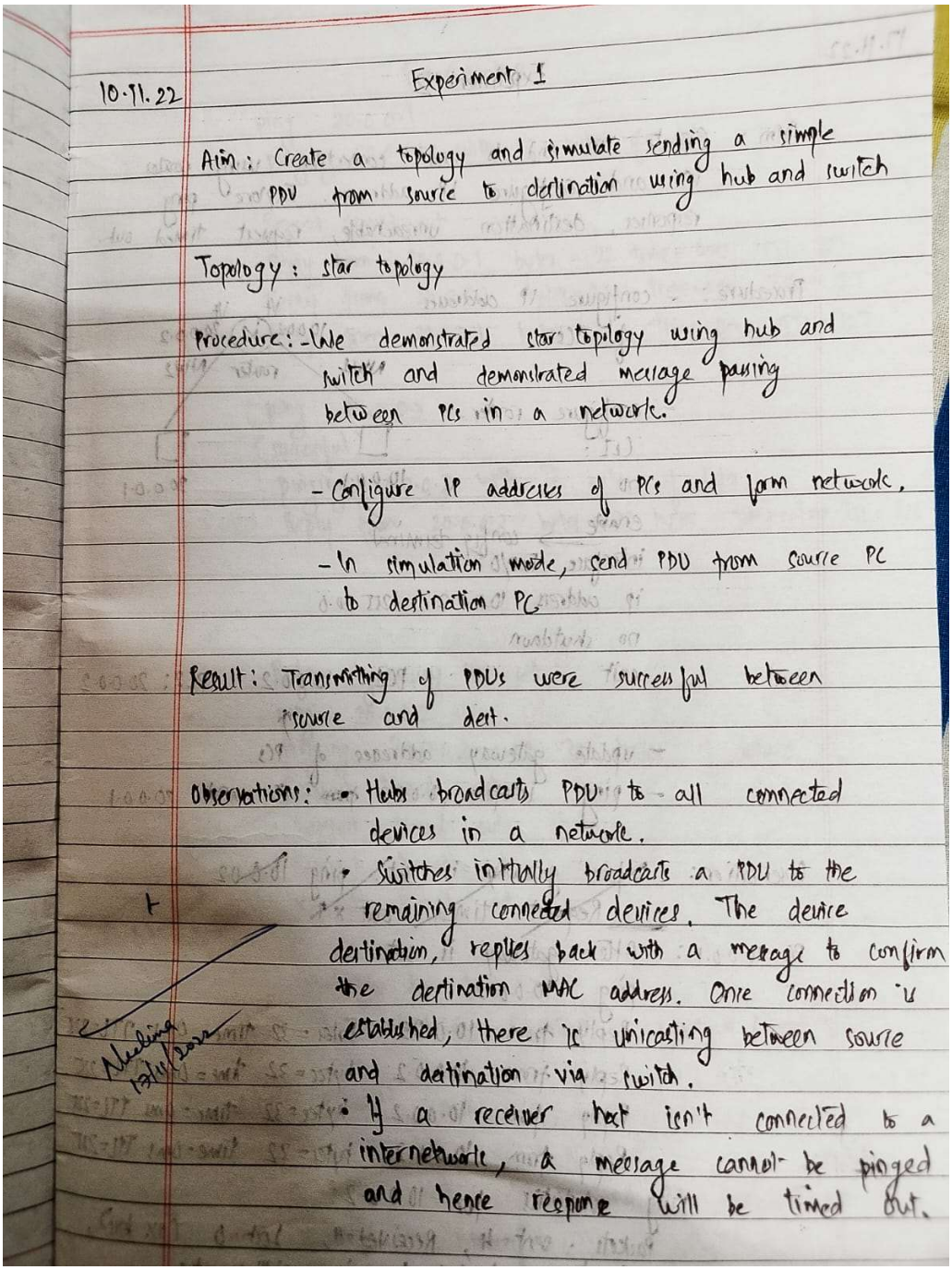
Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

`

Index

Experiment No 1

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



10.11.22

Experiment 1

Aim: Create a topology and simulate sending a simple PDU from source to destination using hub and switch

Topology: star topology

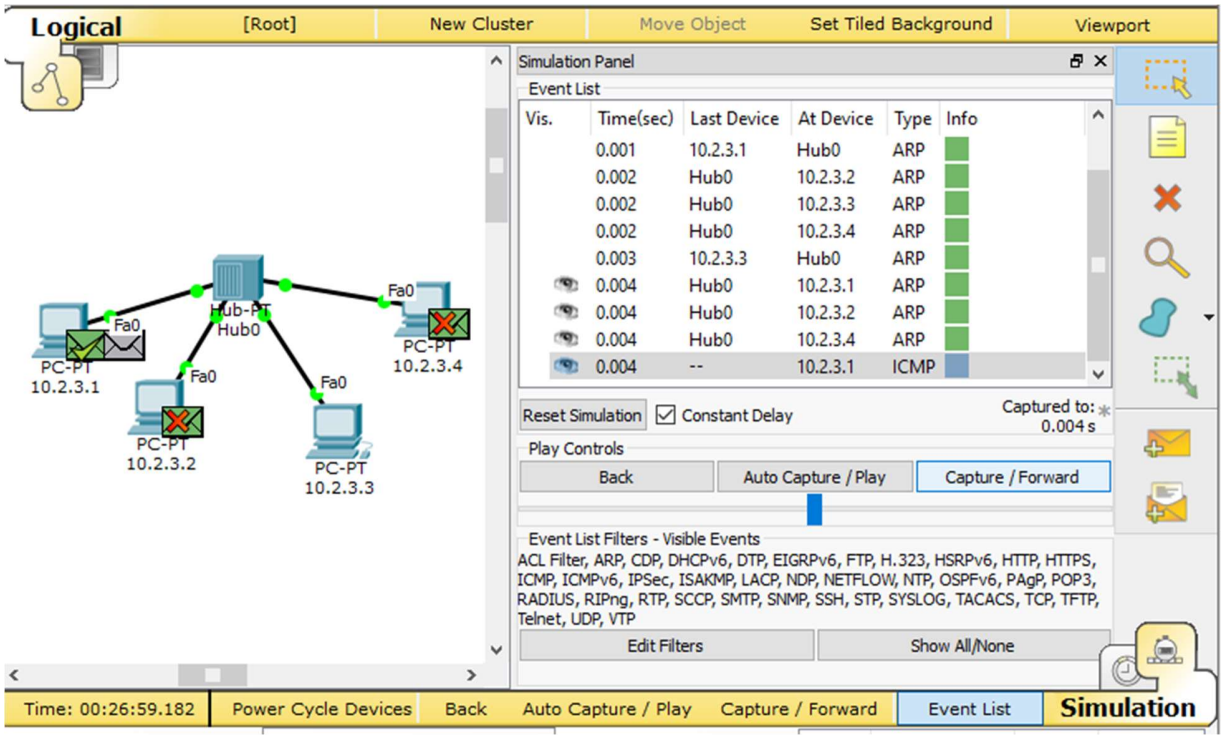
Procedure: - We demonstrated star topology using hub and switch and demonstrated message passing between PCs in a network.

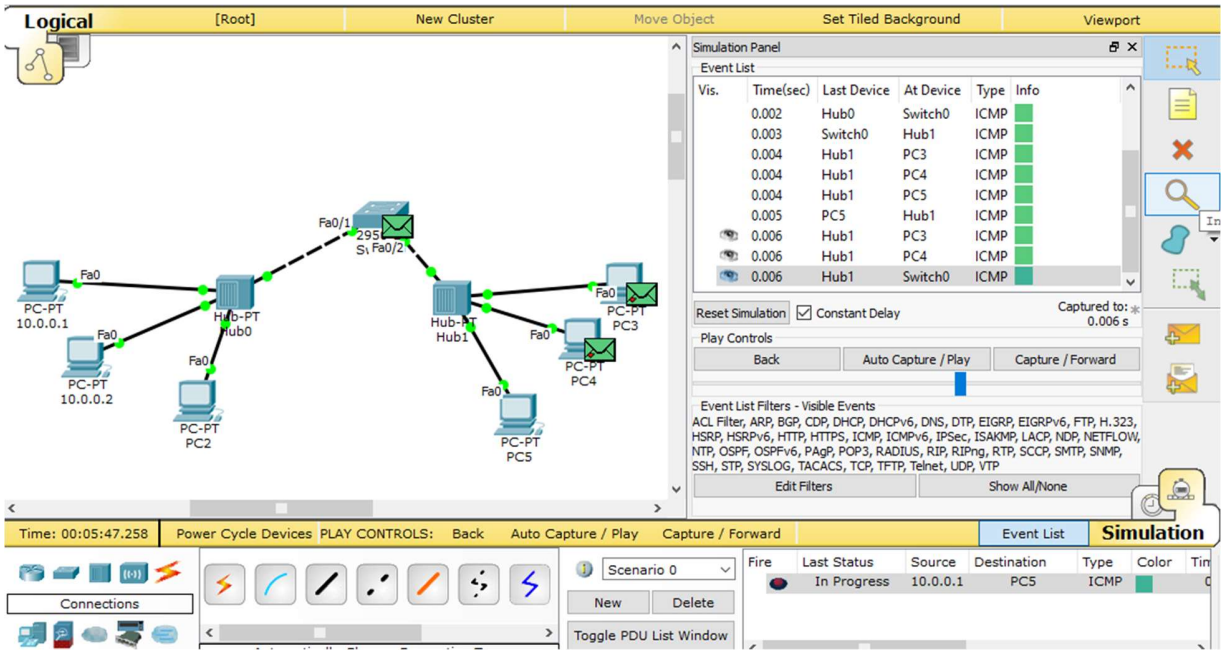
- Configure IP addresses of PCs and form network.

- In simulation mode, send PDU from source PC to destination PC.

Result: Transmitting of PDUs were successful between source and dest.

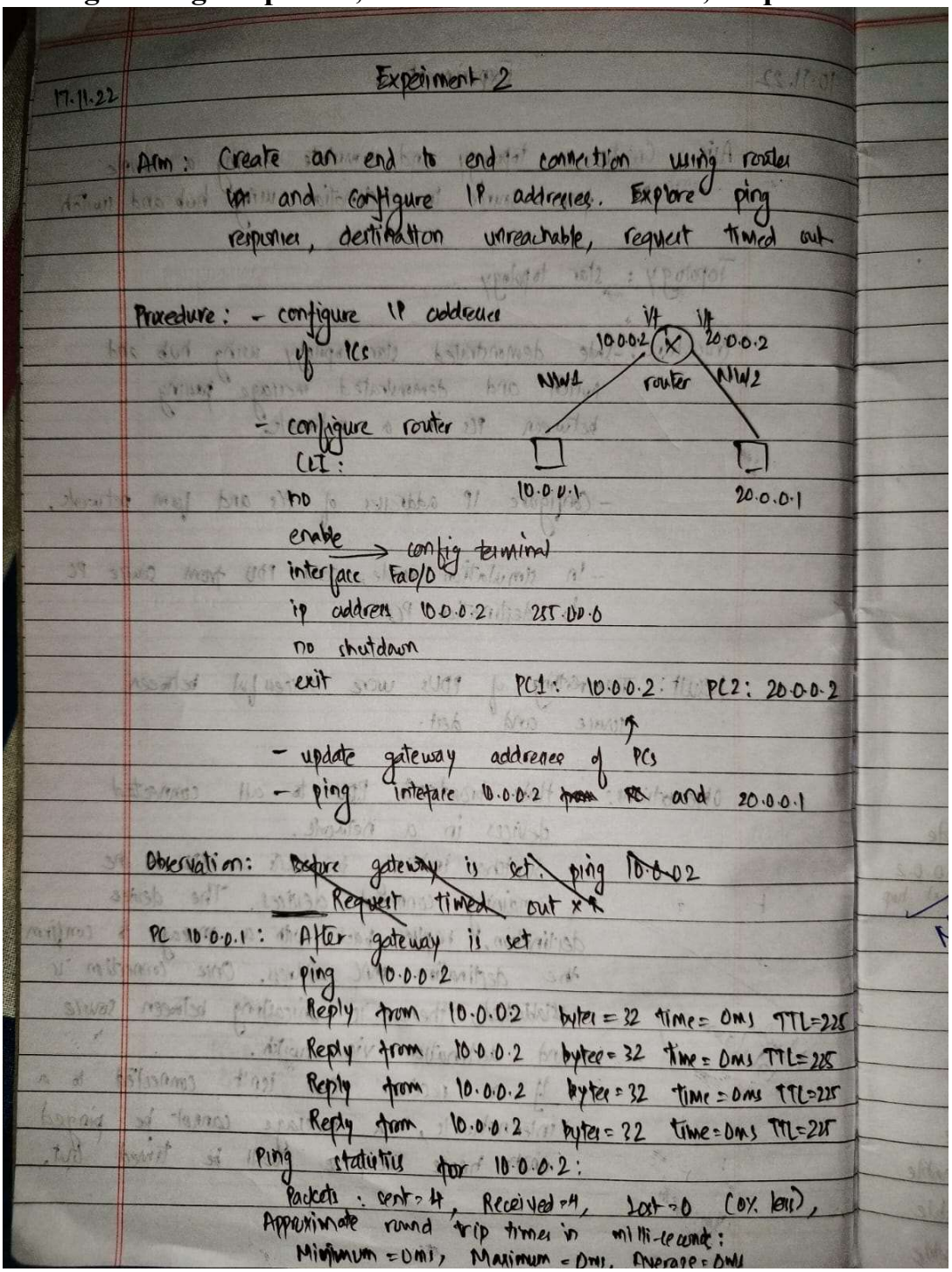
Observations:
- Hubs broadcasts PDU to all connected devices in a network.
- Switches initially broadcasts a PDU to the remaining connected devices. The device destination, replies back with a message to confirm the destination MAC address. Once connection is established, there is unicasting between source and destination via switch.
- If a receiver host isn't connected to a internetwork, a message cannot be pinged and hence response will be timed out.


Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
Simulation Panel
Event List
Vis. Time(sec) Last Device At Device Type Info
0.001 10.2.3.1 Hub0 ARP
0.002 Hub0 10.2.3.2 ARP
0.002 Hub0 10.2.3.3 ARP
0.002 Hub0 10.2.3.4 ARP
0.003 10.2.3.3 Hub0 ARP
0.004 Hub0 10.2.3.1 ARP
0.004 Hub0 10.2.3.2 ARP
0.004 Hub0 10.2.3.4 ARP
0.004 -- 10.2.3.1 ICMP
Reset Simulation
Constant Delay
Captured to: 0.004 s
Play Controls
Back
Auto Capture / Play
Capture / Forward
Event List Filters - Visible Events
ACL Filter, ARP, CDP, DHCPv6, DTP, EIGRPv6, FTP, H.323, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NDP, NETFLOW, NTP, OSPFv6, PAgP, POP3, RADIUS, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP
Edit Filters
Show All/None
Hub-PT Hub0
PC-PT 10.2.3.1
PC-PT 10.2.3.2
PC-PT 10.2.3.3
PC-PT 10.2.3.4
Fa0
Time: 00:26:59.182
Power Cycle Devices
Back
Auto Capture / Play
Capture / Forward
Event List
Simulation


Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
Simulation Panel
Event List
Vis. Time(sec) Last Device At Device Type Info
0.002 Hub0 Switch0 ICMP
0.003 Switch0 Hub1 ICMP
0.004 Hub1 PC3 ICMP
0.004 Hub1 PC4 ICMP
0.004 Hub1 PC5 ICMP
0.005 PC5 Hub1 ICMP
0.006 Hub1 PC3 ICMP
0.006 Hub1 PC4 ICMP
0.006 Hub1 Switch0 ICMP
Reset Simulation
Constant Delay
Captured to: 0.006 s
Play Controls
Back
Auto Capture / Play
Capture / Forward
Event List Filters - Visible Events
ACL Filter, ARP, BGP, CDP, DHCP, DHCPv6, DNS, DTP, EIGRP, EIGRPv6, FTP, H.323, HSRP, HSRPv6, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NDP, NETFLOW, NTP, OSPF, OSPFv6, PAgP, POP3, RADIUS, RIP, RIPng, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP
Edit Filters
Show All/None
Hub-PT Hub0
Hub-PT Hub1
PC-PT 10.0.0.1
PC-PT 10.0.0.2
PC-PT PC2
PC-PT PC3
PC-PT PC4
PC-PT PC5
Fa0
Fa0/1
Fa0/2
Time: 00:05:47.258
Power Cycle Devices
PLAY CONTROLS:
Back
Auto Capture / Play
Capture / Forward
Event List
Simulation
Connections
Scenario 0
New
Delete
Toggle PDU List Window
Fire Last Status Source Destination Type Color
In Progress 10.0.0.1 PC5 ICMP

Experiment No 2

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



Experiment 3

ping 20.0.0.1
pinging 20.0.0.1 with 32 bytes of data:

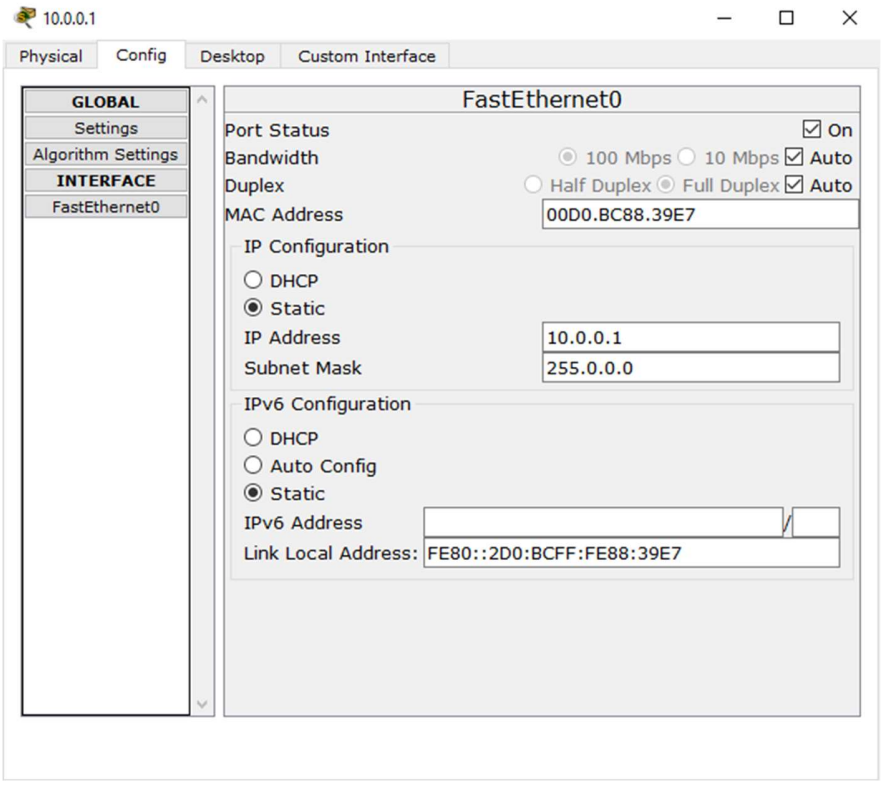
Request timed out
Reply from 20.0.0.1 bytes = 32 time = 0ms TTL = 127
Reply from 20.0.0.1 bytes = 32 time = 0ms TTL = 127
Reply from 20.0.0.1 bytes = 32 time = 0ms TTL = 127

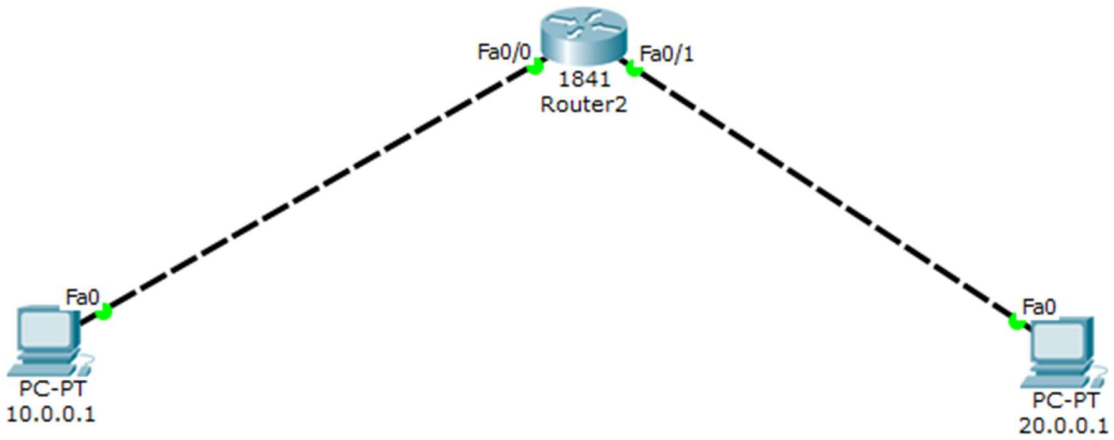
ping 20.0.0.2
(successful)
pinging 20.0.0.2 with 32 bytes of data:
Reply from 20.0.0.2 bytes = 32 bytes time = 1ms TTL = 225

Before gateway is set
ping 20.0.0.2
Pinging 20.0.0.2 with 32 bytes of data:
Request timed out
Request timed out
Request timed out
Request timed out

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

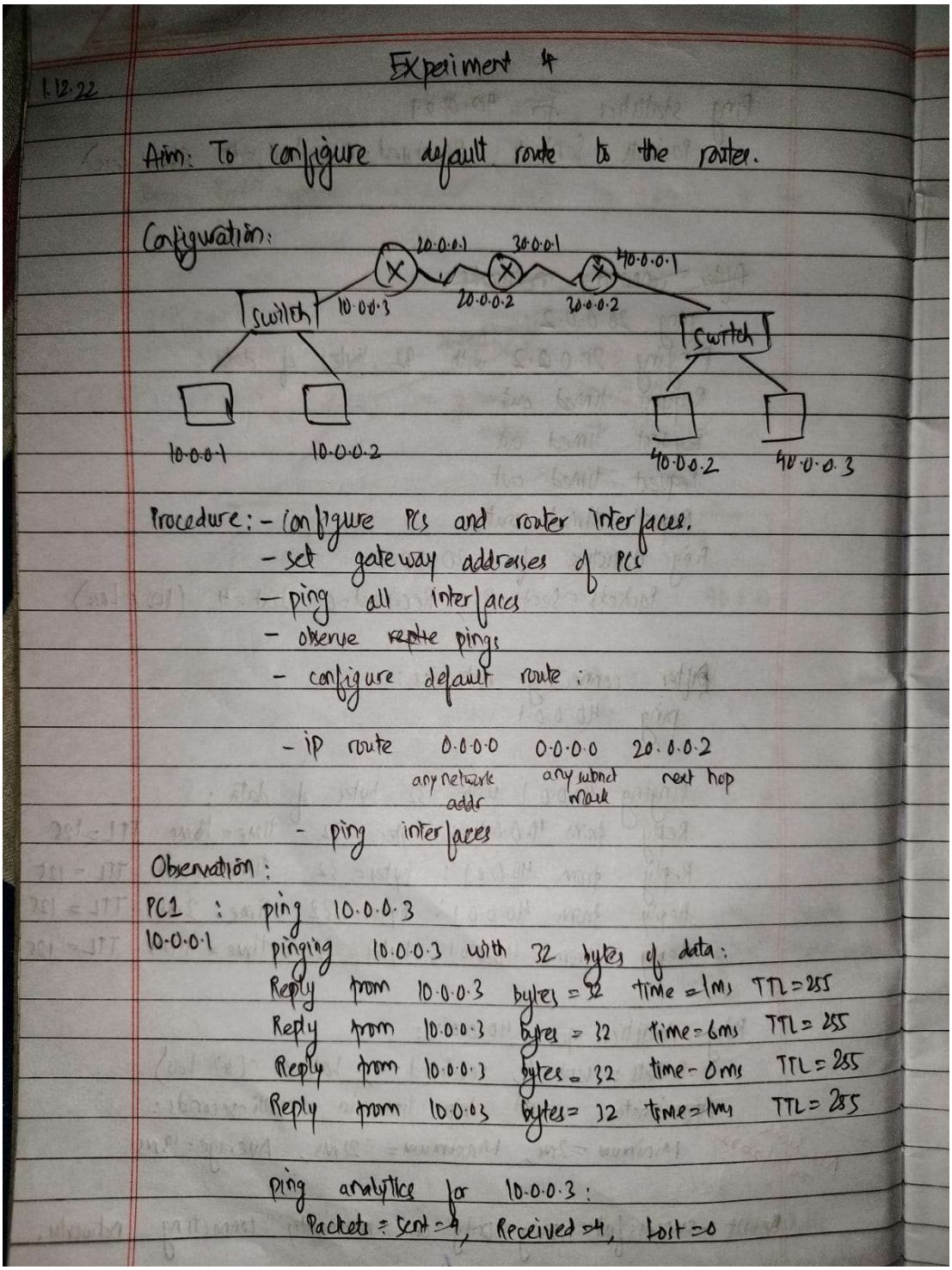
Result: - Request timed out when gateway address is not set
- When both gateway and IP addresses are set, ping is successful.


10.0.0.1
Physical
Config
Desktop
Custom Interface
GLOBAL
Settings
Algorithm Settings
INTERFACE
FastEthernet0
FastEthernet0
Port Status
On
Bandwidth
100 Mbps
10 Mbps
Auto
Duplex
Half Duplex
Full Duplex
Auto
MAC Address
00D0.BC88.39E7
IP Configuration
DHCP
Static
IP Address
10.0.0.1
Subnet Mask
255.0.0.0
IPv6 Configuration
DHCP
Auto Config
Static
IPv6 Address
Link Local Address:
FE80::2D0:BCFF:FE88:39E7


Fa0/0
Fa0/1
1841
Router2
Fa0
PC-PT
10.0.0.1
Fa0
PC-PT
20.0.0.1

Experiment No 3

Configuring default route to the Router



1.12.22

Experiment 4

Aim: To configure default route to the router.

Configuration:

Procedure: - configure PCs and router interfaces.
- set gateway addresses of PCs
- ping all interfaces
- observe route pings
- configure default route:

- ip route 0.0.0.0 0.0.0.0 20.0.0.2
  (0.0.0.0: any network addr; 0.0.0.0: any subnet mask; 20.0.0.2: next hop)
- ping interfaces

Observation:

PC1 : ping 10.0.0.3

10.0.0.1 pinging 10.0.0.3 with 32 bytes of data:

Reply from 10.0.0.3 bytes = 32 time = 1ms TTL = 255
Reply from 10.0.0.3 bytes = 32 time = 6ms TTL = 255
Reply from 10.0.0.3 bytes = 32 time = 0ms TTL = 255
Reply from 10.0.0.3 bytes = 32 time = 1ms TTL = 255

ping analytics for 10.0.0.3:
Packets: Sent = 4, Received = 4, Lost = 0

ping 20.0.0.1

pinging 20.0.0.1 with 32 bytes of data:

Reply from 20.0.0.1 : bytes = 32 time=0ms TTL=255

Reply from 20.0.0.1 : bytes = 32 time=5ms TTL = 255

Reply from 20.0.01 : bytes = 32 time=0ms TTL = 255

Reply from 20.0.0.1 : bytes = 32 time=0ms TTL = 255

ping statistics from 20.0.0.1:

packets: sent=4, Received=4, Lost=0

ping 20.0.0.2

pinging 20.0.0.2 with 32 bytes of data:

Request timed out

Request timed out

Request timed out

Request timed out

ping statistics for 20.0.0.2:

packets: sent=4, Received=0, Lost=4 (100% loss)

ping 30.0.0.1

pinging 30.0.0.1 with 32 bytes of data:

Reply from 10.0.0.3

Reply from 10.0.0.3

Reply from 10.0.0.3

Reply from 10.0.0.3

After setting default routes:

ping 30.0.0.1

pinging 30.0.0.1 with 32 bytes of data:

Request timed out

Reply from 30.0.0.1 : bytes = 32 time = 1ms TTL = 254

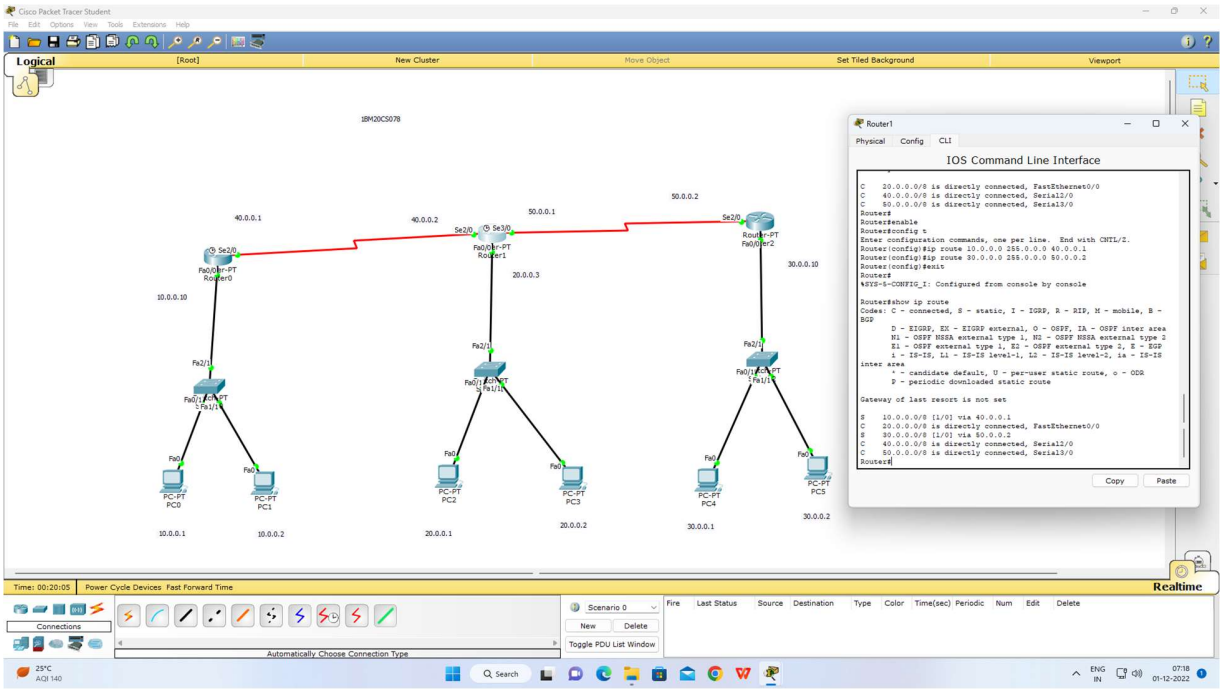
Request timed out
Reply from 30.0.0.1 : bytes = 32 time = 1ms TTL = 254

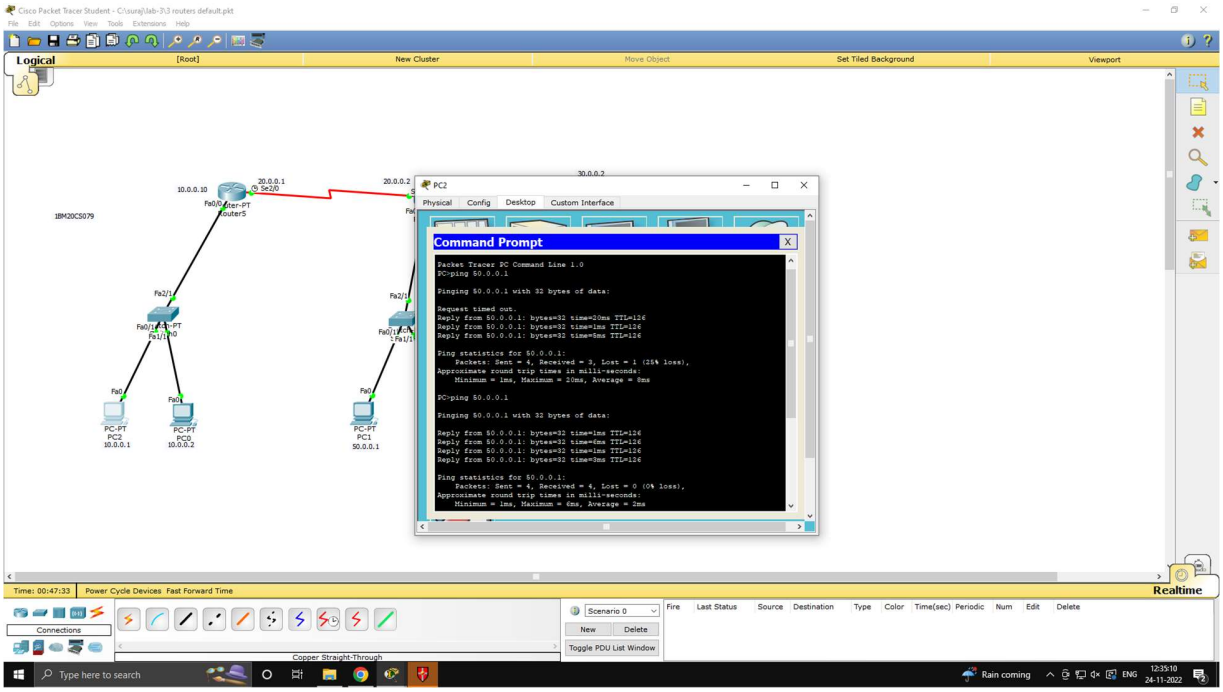
Ping statistics for 30.0.0.1 :
packets : sent = 4 , received = 2 , lost = 2 (50% loss)

ping 40.0.0.2
ping 40.0.0.2 with 32 bytes of data :
Reply from 40.0.0.2 : bytes = 32 time = 11ms TTL = 125
Reply from 40.0.0.2 : bytes = 32 time = 3ms TTL = 125
Reply from 40.0.0.2 : bytes = 32 time = 3ms TTL = 125
Reply from 40.0.0.2 : bytes = 32 time = 3ms TTL = 125

ping statistics for 40.0.0.2 :
packets : sent = 4, received = 4 , lost = 0 (0% loss)

Neelima
1/12/2022


Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
1BM20CS078
Router1
Physical
Config
CLI
IOS Command Line Interface
Router#show ip route
Router#enable
Router#config t
Router(config)#ip route 10.0.0.0 255.0.0.0 40.0.0.1
Router(config)#ip route 30.0.0.0 255.0.0.0 50.0.0.2
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console
Copy
Paste
Router0
Router1
Router2
PC0
PC1
PC2
PC3
PC4
PC5
Time: 00:20:05
Power Cycle Devices
Fast Forward Time
Realtime
Connections
Automatically Choose Connection Type
Scenario 0
New
Delete
Toggle PDU List Window


Logical
[Root]
New Cluster
Move Object
Set Tiled Background
Viewport
1BM20CS079
PC2
Physical
Config
Desktop
Custom Interface
Command Prompt
Packet Tracer PC Command Line 1.0
PC>ping 50.0.0.1
Pinging 50.0.0.1 with 32 bytes of data:
Request timed out.
Reply from 50.0.0.1: bytes=32 time=20ms TTL=126
Reply from 50.0.0.1: bytes=32 time=1ms TTL=126
Reply from 50.0.0.1: bytes=32 time=8ms TTL=126
Ping statistics for 50.0.0.1:
Packets: Sent = 4, Received = 3, Lost = 1 (25% loss),
Approximate round trip times in milli-seconds:
Minimum = 1ms, Maximum = 20ms, Average = 9ms
PC>ping 50.0.0.1
Pinging 50.0.0.1 with 32 bytes of data:
Reply from 50.0.0.1: bytes=32 time=1ms TTL=126
Reply from 50.0.0.1: bytes=32 time=4ms TTL=126
Reply from 50.0.0.1: bytes=32 time=1ms TTL=126
Reply from 50.0.0.1: bytes=32 time=3ms TTL=126
Ping statistics for 50.0.0.1:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 1ms, Maximum = 4ms, Average = 2ms
Time: 00:47:33
Power Cycle Devices
Fast Forward Time
Realtime
Connections
Copper Straight-Through
Scenario 0
New
Delete
Toggle PDU List Window

Experiment No 4

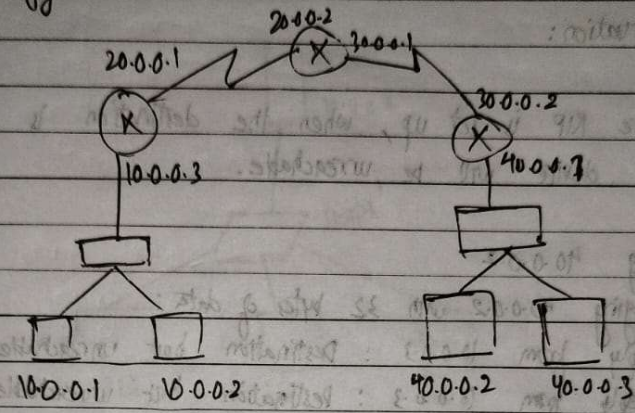
Configuring default route to the Router

8.12.22

Experiment 5

Aim: To configure RIP protocol for routers

Configuration:



Procedure:

- Construct topology and configure ip addresses of end devices and interface
- Configure router interface but with serial interface encapsulate with ppp and set clock rate

at R1: ~~enable~~ ~~interface~~
config ter
interface se2/0
ip address 20.0.0.1 255.0.0.0
encapsulation ppp
clock rate 64000
no shutdown

Once the connections are up, let the router rip for all remaining routers and connect them to the known ~~methodes~~ networks.

Observation :

IPs are dynamically allocated to all PCs

successful pings to PCs

Neelima
15/12/2022

router rip
network 10.0.0.0
network 20.0.0.0
exit

Observation:

before RIP is set up, when the destination is pinged, the device will be unreachable.

ping 40.0.0.2
pinging 40.0.0.2 with 32 bytes of data:
Reply from 10.0.0.3 : Destination host unreachable
Reply from 10.0.0.3 : Destination host unreachable
Reply from 10.0.0.3 : Destination host unreachable
Reply from 10.0.0.3 : Destination host unreachable

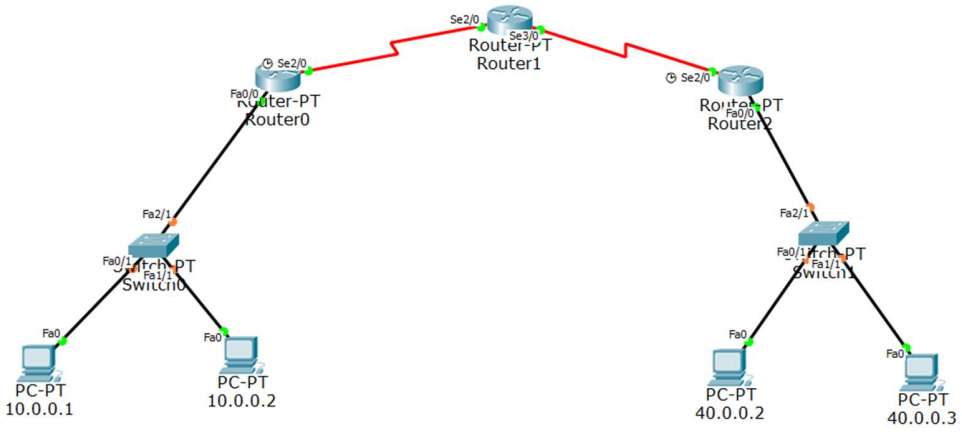
Ping statistics for 40.0.0.2:
Packets: sent=4, Received=0, Lost=4 (100% loss)

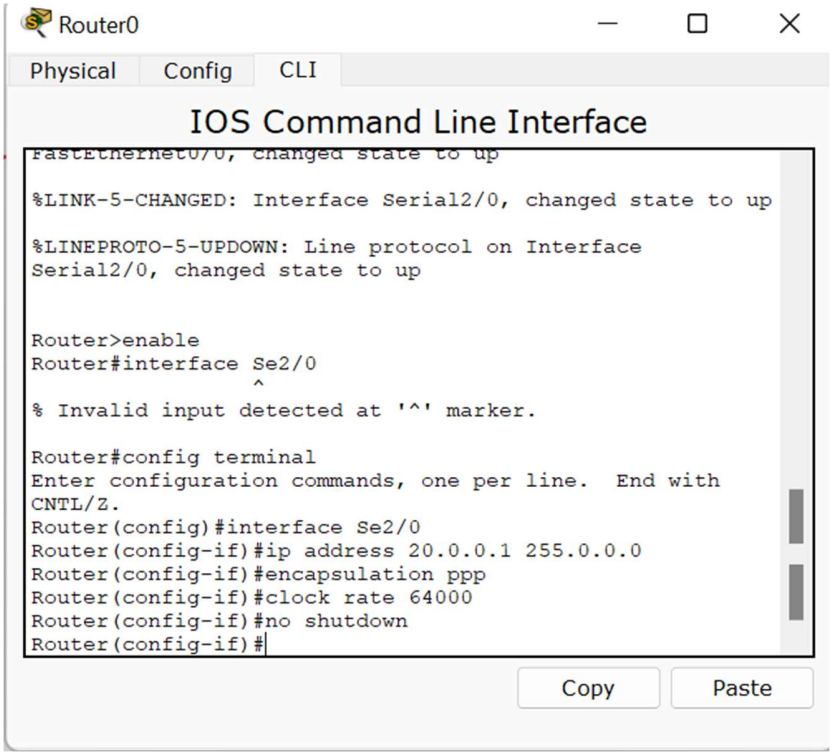
When RIP is set,

ping 40.0.0.2
pinging 40.0.0.2 with 32 bytes of data:
Request Timed out.
Reply from 40.0.0.2 : bytes = 32 time = 7ms TTL=125
Reply from 40.0.0.2 : bytes = 32 time = 6ms TTL = 125
Reply from 40.0.0.2 : bytes = 32 time = 7ms TTL = 125

ping statistics for 40.0.0.2:
Packets: sent=4, Received=3, Lost=1 (25% Loss)

Neelima 8/12/2022


Se2/0
Se3/0
Router-PT
Router1
Se2/0
Fa0/0
Router-PT
Router0
Se2/0
Fa0/0
Router-PT
Router2
Fa2/1
Fa0/1
Fa1/1
Switch0
Fa2/1
Fa0/1
Fa1/1
Switch1
Fa0
Fa0
Fa0
Fa0
PC-PT
10.0.0.1
PC-PT
10.0.0.2
PC-PT
40.0.0.2
PC-PT
40.0.0.3


Router0
Physical
Config
CLI
IOS Command Line Interface
FastEthernet0/0, changed state to up
%LINK-5-CHANGED: Interface Serial2/0, changed state to up
%LINEPROTO-5-UPDOWN: Line protocol on Interface
Serial2/0, changed state to up
Router>enable
Router#interface Se2/0
^
% Invalid input detected at '^' marker.
Router#config terminal
Enter configuration commands, one per line. End with
CNTL/Z.
Router(config)#interface Se2/0
Router(config-if)#ip address 20.0.0.1 255.0.0.0
Router(config-if)#encapsulation ppp
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown
Router(config-if)#
Copy
Paste

Experiment No 5

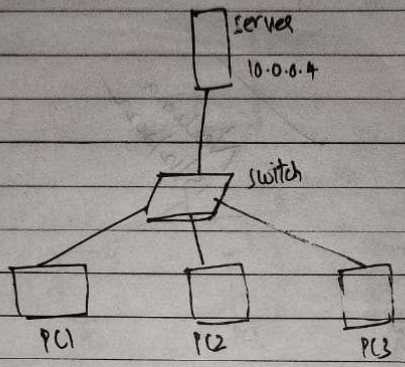
Configuring RIP Routing Protocol in Routers

15.12.22

Experiment-6

Aim : configure DHCP to server

Configuration:

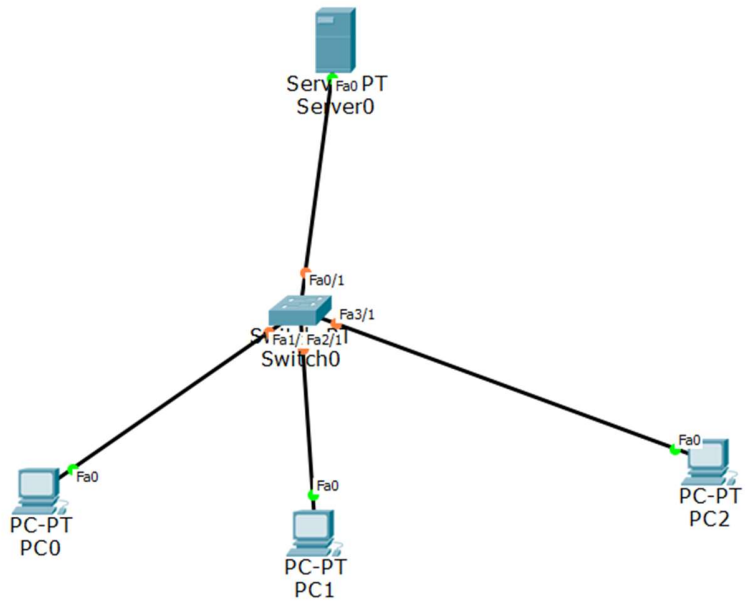


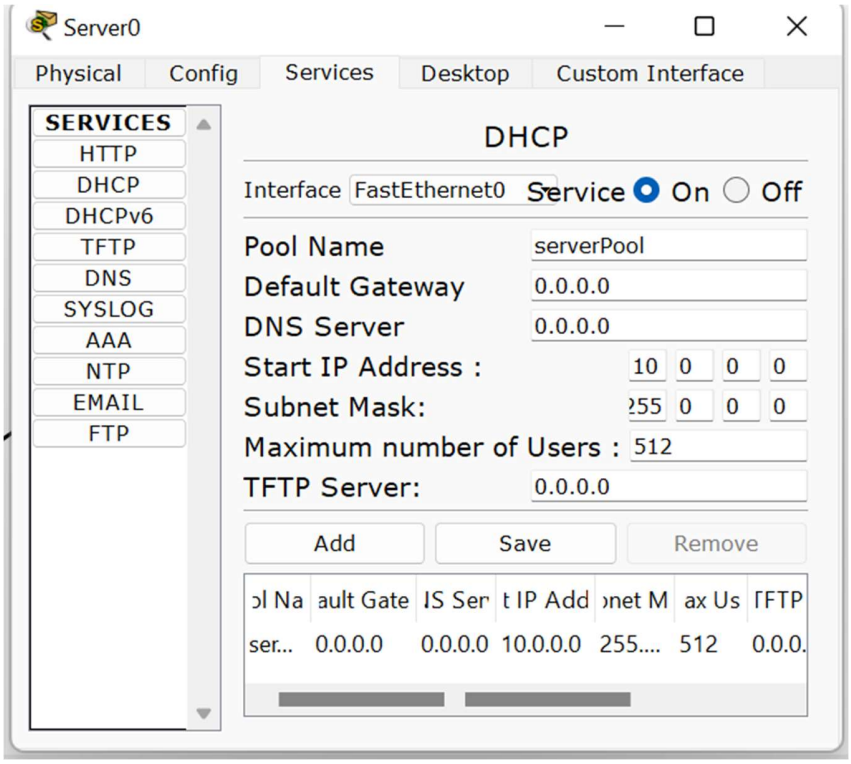
Procedure :

- Construct topology. Configure IP address of server
- Server: under services choose DHCP.
  Turn service on
  set start IP Address : 10.0.0.1
  subnet mask : 255.0.0.0
  save
- ~~PCs are dynamically allocated IP addresses~~
- PCs : change IP config from static to DHCP

-PCs are dynamically allocated IP addresses

- RARP protocol is used


Fa0
Server0
Fa0/1
Fa3/1
Fa1
Fa2/1
Switch0
Fa0
PC-PT
PC0
Fa0
PC-PT
PC1
Fa0
PC-PT
PC2


Server0
Physical
Config
Services
Desktop
Custom Interface
SERVICES
HTTP
DHCP
DHCPv6
TFTP
DNS
SYSLOG
AAA
NTP
EMAIL
FTP
DHCP
Interface
FastEthernet0
Service
On
Off
Pool Name
serverPool
Default Gateway
0.0.0.0
DNS Server
0.0.0.0
Start IP Address :
10 0 0 0
Subnet Mask:
255 0 0 0
Maximum number of Users :
512
TFTP Server:
0.0.0.0
Add
Save
Remove
ol Na
ault Gate
IS Ser
t IP Add
onet M
ax Us
TFTP
ser...
0.0.0.0
0.0.0.0
10.0.0.0
255....
512
0.0.0.

Experiment No 6

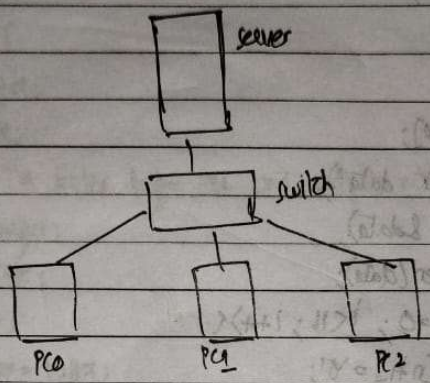
Demonstration of WEB server and DNS using Packet Tracer

15.12.22

Experiment - 7

Aim: How to configure web server and DNS server

Configuration:

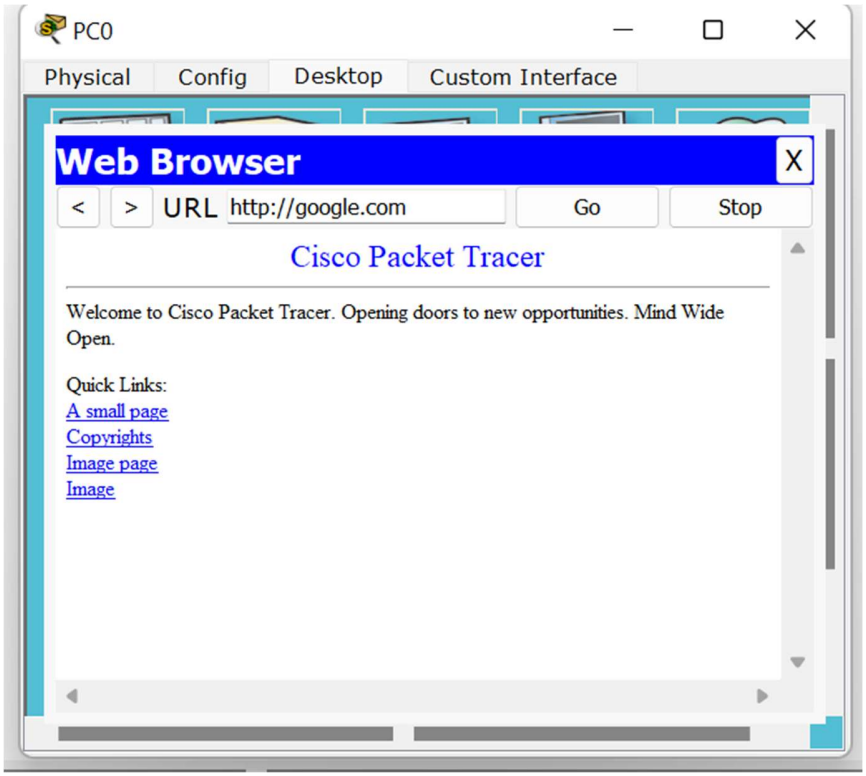


Procedure:

- construct topology
- configure server and end-device IP addresses
- Server: Turn HTTP on
  Turn DNS service on
  Give domain name, address and add
- PC: on web browser,
  give address or domain name to load the page

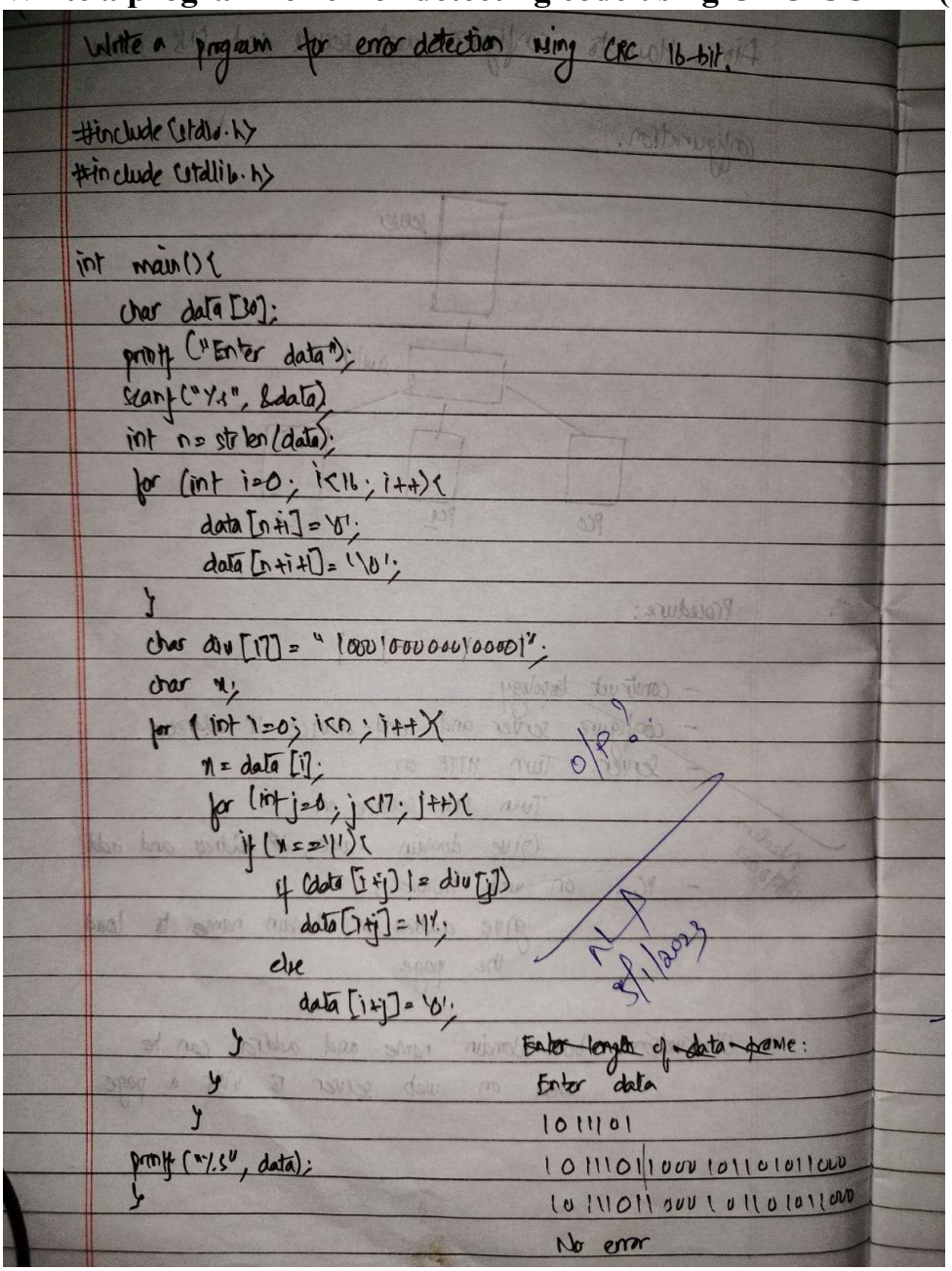
Observation: both domain name and address can be used on web server to visit a page


Fa0
Server-PT
Server0
Fa0
PC-PT
PC0


PC0
Physical
Config
Desktop
Custom Interface
Web Browser
X
<
>
URL
http://google.com
Go
Stop
Cisco Packet Tracer
Welcome to Cisco Packet Tracer. Opening doors to new opportunities. Mind Wide Open.
Quick Links:
A small page
Copyrights
Image page
Image

Experiment No 7

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



Write a program for error detection using CRC 16-bit.

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

int main(){
    char data[30];
    printf("Enter data");
    scanf("%s", &data)
    int n = strlen(data);
    for(int i=0; i<16; i++){
        data[n+i] = '0';
        data[n+i+1] = '\0';
    }
    char div[17] = "10001000000100001";
    char x;
    for(int i=0; i<n; i++){
        x = data[i];
        for(int j=0; j<17; j++){
            if(x=='1'){
                if(data[i+j] != div[j])
                data[i+j] = '1';
                else
                data[i+j] = '0';
            }
        }
    }
    printf("%s", data);
}
```

o/p:

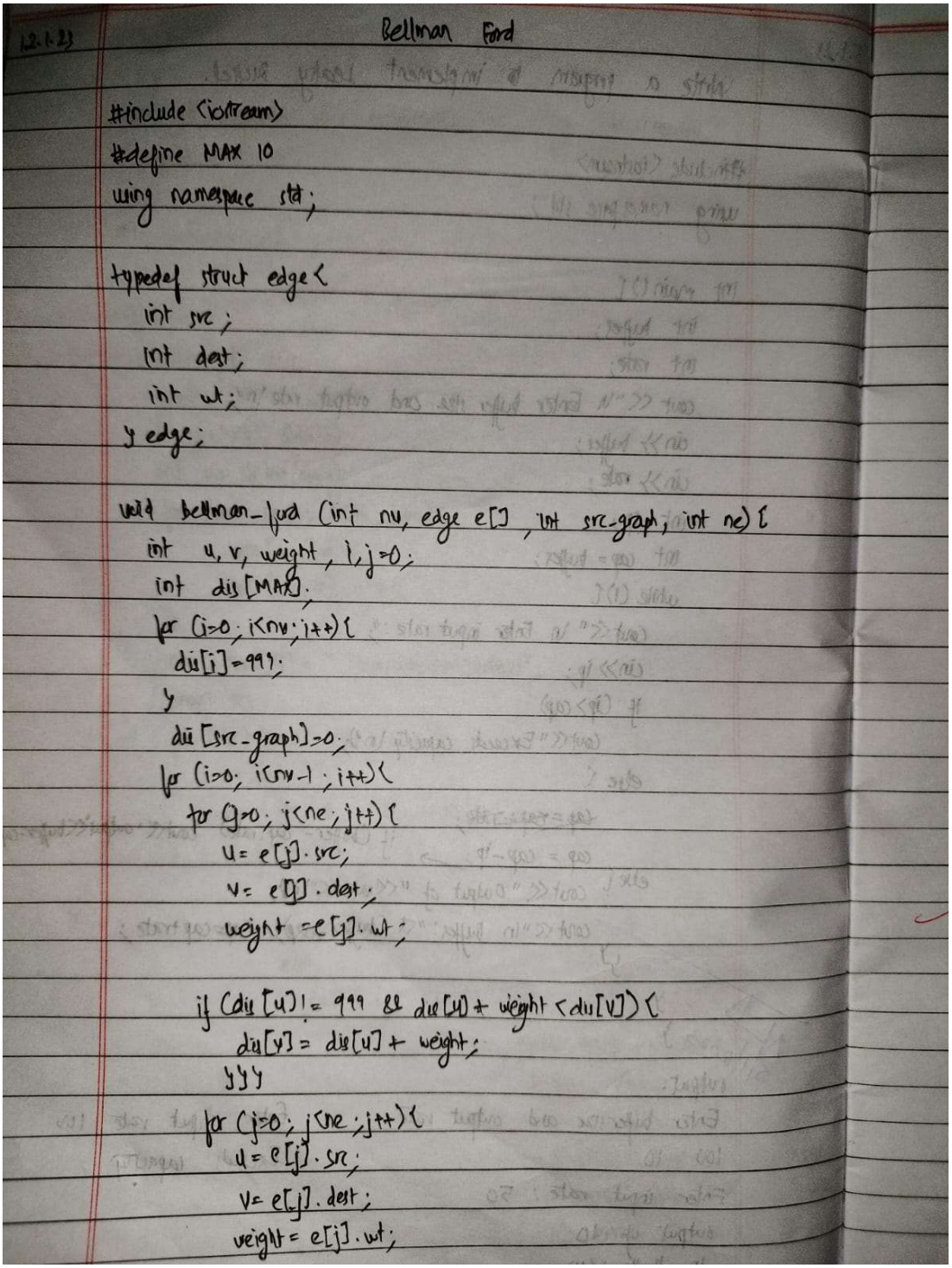
```
Enter length of data frame:
Enter data
1011101
1011101|0000 1011010111000
1011011 0001 0110101 1000
No error
```

21/1/2023

```
enter data
1011101

Original data :1011101
G(x) :10001000000100001

Sender
Modified Data :10111010000000000000000
Checksum :1000101101011000

Receiver
Codeword :10111011000101101011000
Checksum :0000000000000000
```

Experiment No 8

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



Bellman Ford

```
#include <iostream>
#define MAX 10
using namespace std;

typedef struct edge{
  int src;
  int dest;
  int wt;
}edge;

void bellman_ford (int nv, edge e[], int src_graph, int ne){
  int u, v, weight, i, j=0;
  int dis[MAX];
  for (i=0; i<nv; i++){
    dis[i]=999;
  }
  dis[src_graph]=0;
  for (i=0; i<nv-1; i++){
    for (j=0; j<ne; j++){
      u= e[j].src;
      v= e[j].dest;
      weight =e[j].wt;

      if (dis[u]!= 999 && dis[u]+ weight <dis[v]){
        dis[v]= dis[u]+ weight;
      }}}
    for (j=0; j<ne; j++){
      u= e[j].src;
      v= e[j].dest;
      weight= e[j].wt;
```

```
if (dis[u] + weight < dis[v]){
    cout << "\n| Negative cycle present \n";
    return;
}

cout << "\n Vertex " << " Distance from source ";
for (i=1; i<=n; i++) {
    cout << "\n" << i << "\t" << dis[i];
}

int main() {
    int nv, ne, src_graph;
    edge e[MAX];
    cout << " Enter number of vertices ";
    cin >> nv;
    printf ("Enter source vertex of graph");
    cin >> src_graph;
    cout << "No of edges:";
    cin >> ne

    for (int i=0; i<ne; i++){
        cout << "\n For edge" << i+1 << " => ";
        cout << "\n Enter source vertex ";
        cin >> e[i].src;
        cout << " Enter dest ";
        cin >> e[i].dest;
        cout << " Enter weight ";
        cin >> e[i].wt;
    }
    bellman_ford (nv, e, src_graph, ne);
}
```

Output:

4 vertices

source vertex: 1

edge 1:
SV = 1
DV = 2
wt = 4

edge 2:
SV = 1
DV = 3
wt = 3

edge 3:
SV = 2
DV = 4
wt = 7

edge 4:
SV = 3
DV = 4
wt = 2

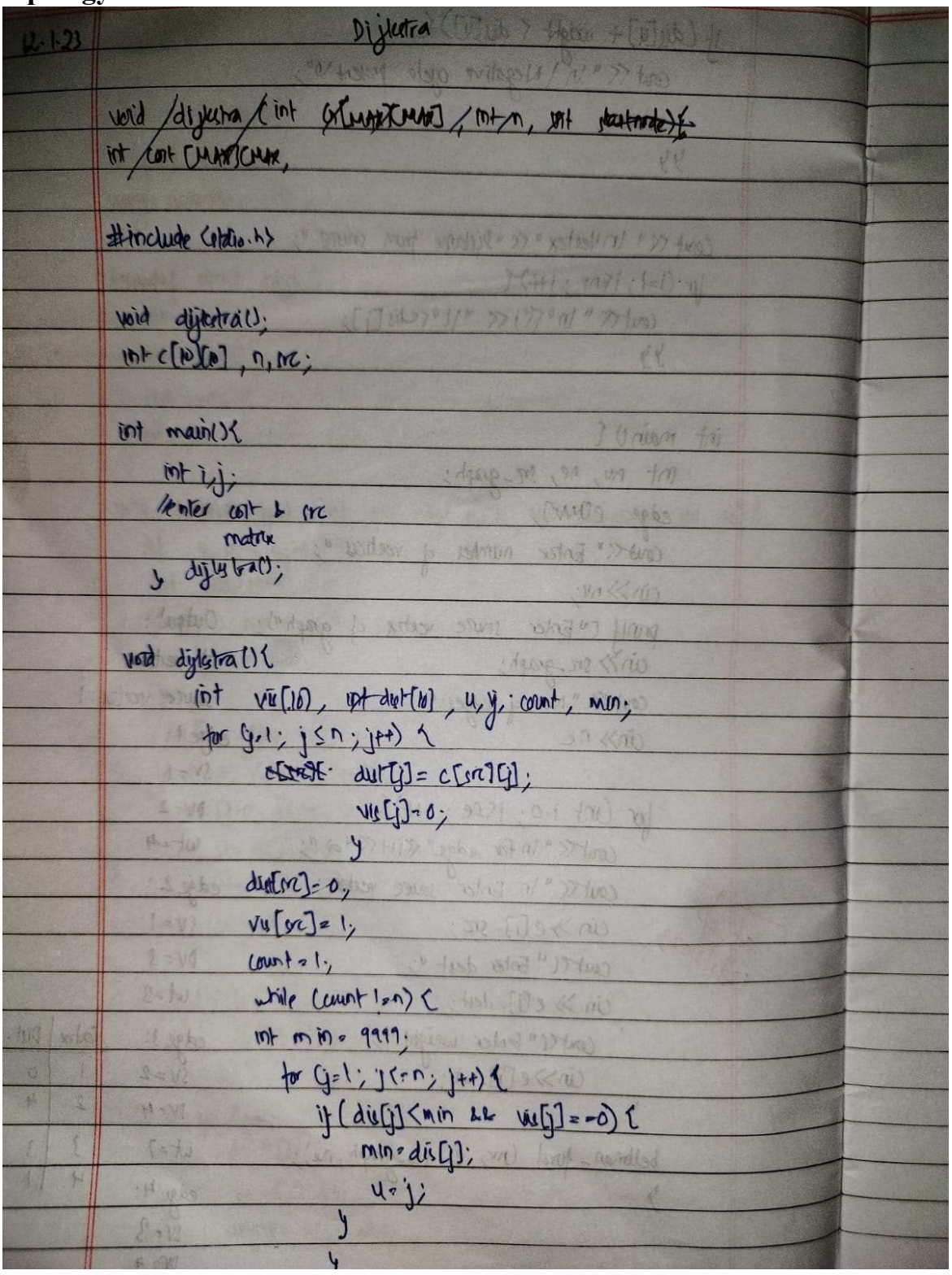
Vertex	Dist.
1	0
2	4
3	3
4	1

```
Enter the number of nodes : 4

Enter the cost matrix :
0 8 999 5
8 0 2 999
999 2 0 3
5 999 3 0


 For router 1

node 1 via 1 Distance 0
node 2 via 2 Distance 8
node 3 via 4 Distance 8
node 4 via 4 Distance 5

 For router 2

node 1 via 1 Distance 8
node 2 via 2 Distance 0
node 3 via 3 Distance 2
node 4 via 3 Distance 5
```

```
 For router 3

node 1 via 4 Distance 8
node 2 via 2 Distance 2
node 3 via 3 Distance 0
node 4 via 4 Distance 3

 For router 4

node 1 via 1 Distance 5
node 2 via 3 Distance 5
node 3 via 3 Distance 3
node 4 via 4 Distance 0
```

Experiment No 9

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



```
Dijkstra

void dijkstra(int [][], int n, int [][]);

#include <stdio.h>

void dijkstra();
int c[10][10], n, src;

int main(){
    int i,j;
    //enter cost & src
        matrix
    dijkstra();

void dijkstra(){
    int vis[10], dist[10], u, j, count, min;
    for(j=1; j<=n; j++){
        dist[j] = c[src][j];
        vis[j]=0;
    }
    dist[src]=0;
    vis[src]=1;
    count=1;
    while(count!=n){
    int min=9999;
      for(j=1; j<=n; j++){
        if(dis[j]<min && vis[j]==0){
            min=dis[j];
            u=j;
        }
    }
```

```
vis[u]=1;
count++;
for(j=1; j<n; j++){
    if(min+c[u][j] < dis[j] && vis[j]==0){
        dis[j] = min+c[u][j];
    }
}
}
printf("shortest distance\n:");
for(j=1; j<n; j++){
    printf("\n%d ... >%d=%d", src, j, dis[j]);
}
}
```

enter no of vertices = 4

cost matrix =

999	9	2	5
9	999	6	8
2	6	999	999
5	8	999	999

~~vertex~~

1 ... > 1 = 0

1 ... > 2 = 8

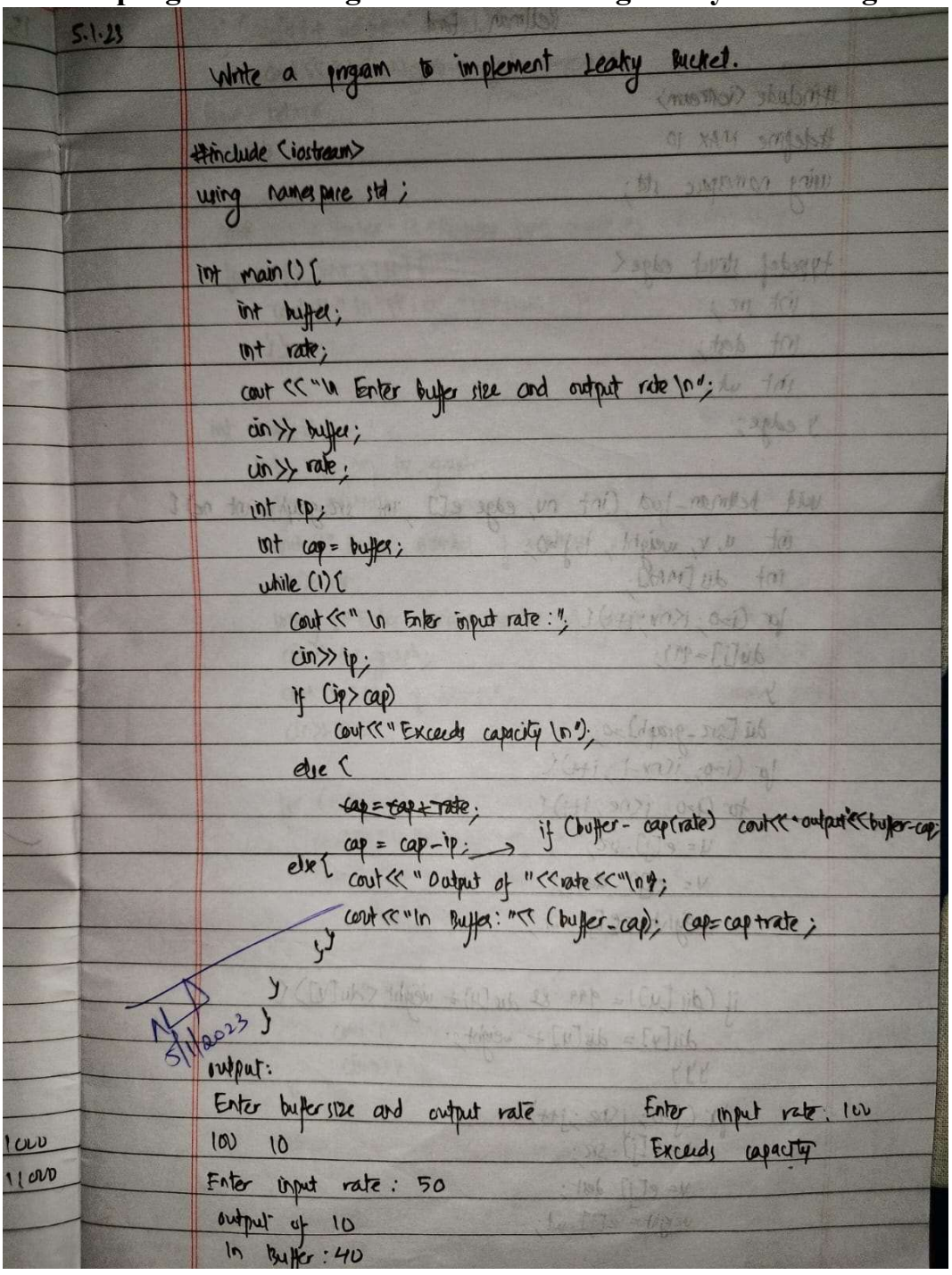
1 ... > 3 = 2

1 ... > 4 = 5

```
Enter the num of vertices:

Enter the cost matrix:
999 9 2 5
9 999 6 8
2 6 999 999
5 8 999 999

Enter the source node:  1

The shortest distance is:

1----->1 = 0
1----->2 = 8
1----->3 = 2
1----->4 = 5
```

Experiment No 10

Write a program for congestion control using Leaky bucket algorithm.



5.1.23

Write a program to implement Leaky Bucket.

```
#include <iostream>
using namespace std;

int main(){
    int buffer;
    int rate;
    cout<<"\n Enter buffer size and output rate \n";
    cin>>buffer;
    cin>>rate;
    int ip;
    int cap = buffer;
    while(1){
        cout<<"\n Enter input rate :";
        cin>>ip;
        if (ip>cap)
            cout<<"Exceeds capacity \n";
        else {
            cap=cap+rate;
            cap = cap-ip;   -> if (buffer - cap(rate) cout<<"output"<<buffer-cap;
        else{ cout<<"Output of "<<rate<<"\n";
            cout<<"\n Buffer: "<< (buffer-cap); cap=cap+rate;
        }
    }
}
```

5/1/2023

output:

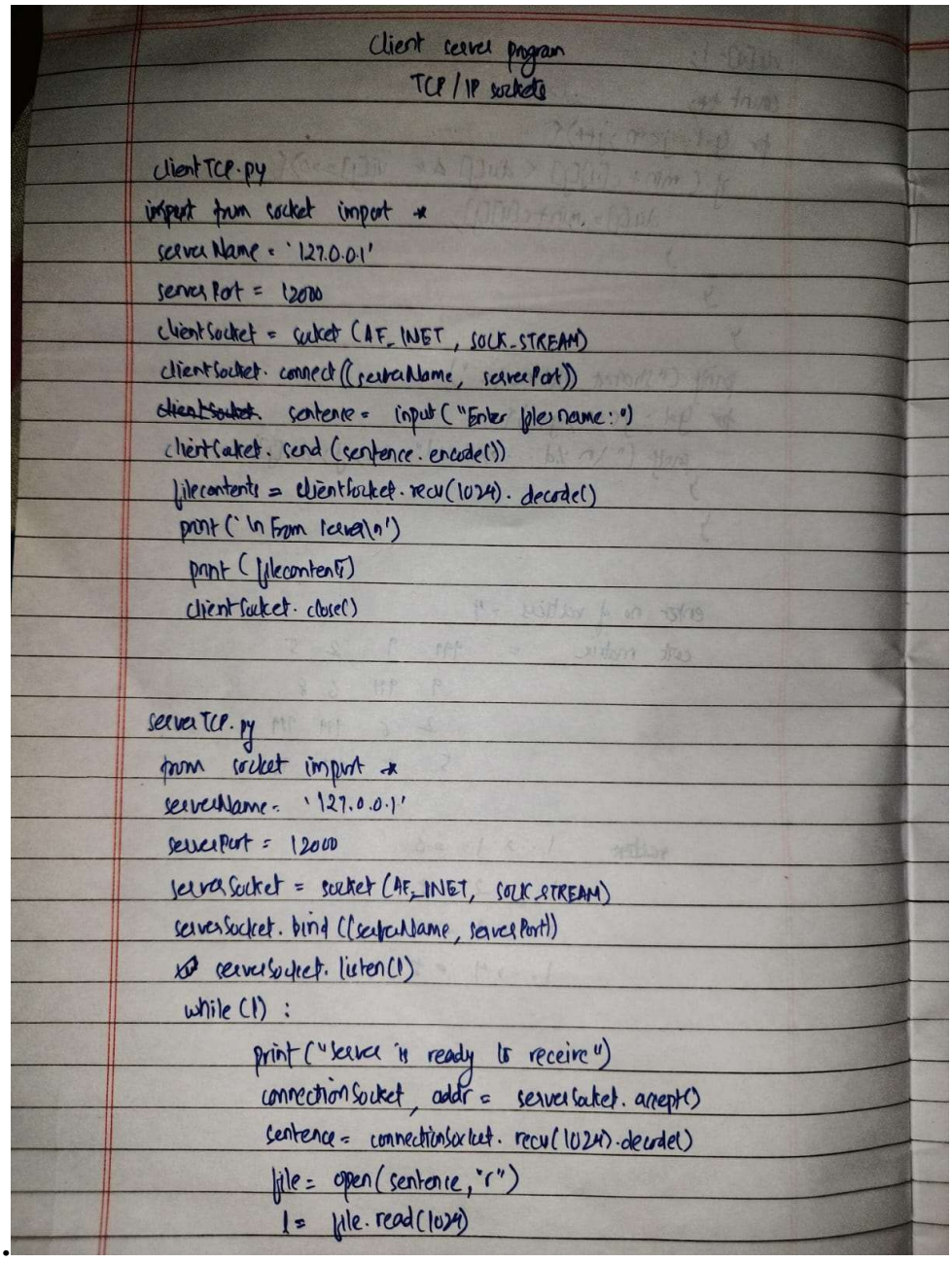
Enter buffersize and output rate
100 10
Enter input rate : 50
output of 10
In Buffer : 40

Enter input rate: 101
Exceeds capacity

```
Enter buffer size and output rate
100
10

Enter input rate:50
Output of 10
In Buffer:40
Enter input rate:70
Exceeds capacity
```

Experiment No 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.



Client server program
TCP / IP sockets

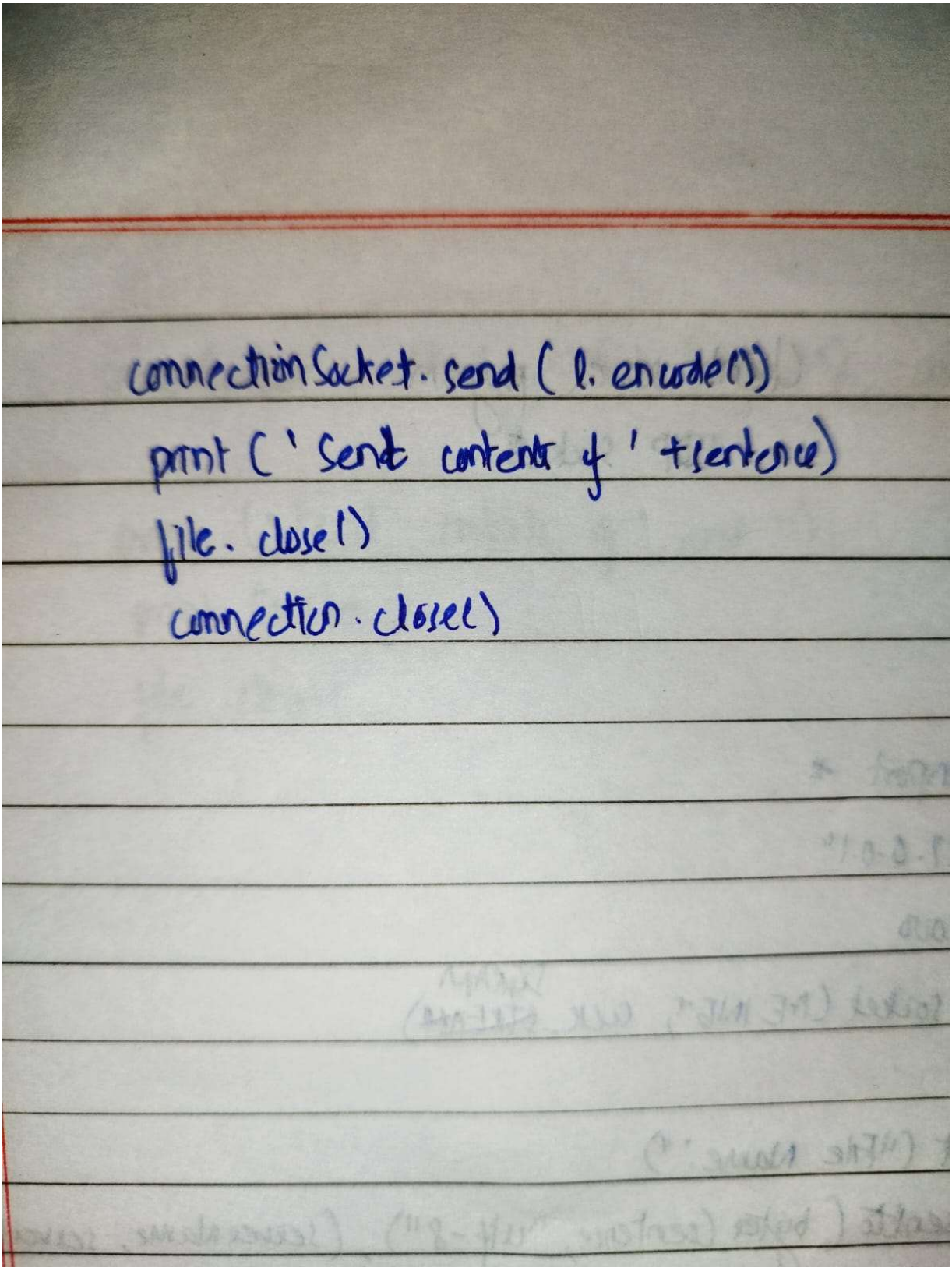
clientTCP.py

```
import from socket import *
serverName = '127.0.0.1'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort))
clientSocket. sentence = input("Enter file name:")
clientSocket.send(sentence.encode())
filecontents = clientSocket.recv(1024).decode()
print('\n From receiver')
print(filecontents)
clientSocket.close()
```

serverTCP.py

```
from socket import *
serverName = '127.0.0.1'
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort))
serverSocket.listen(1)
while(1):
    print("server is ready to receive")
    connectionSocket, addr = serverSocket.accept()
    sentence = connectionSocket.recv(1024).decode()
    file = open(sentence, "r")
    l = file.read(1024)
```

.

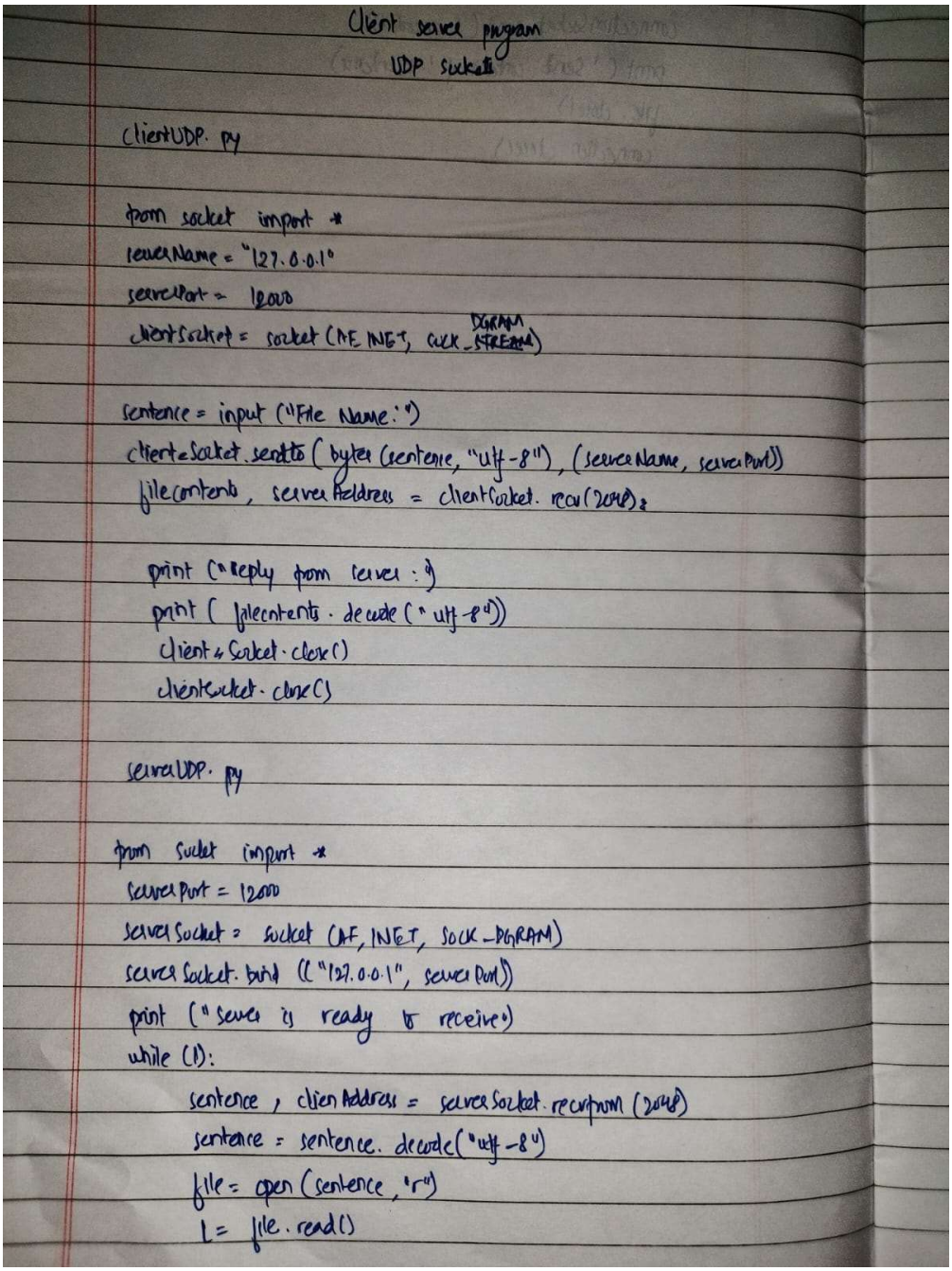

connectionSocket.send(l.encode())
print('Sent contents of '+sentence)
file.close()
connection.close()

```
Enter file name: serverTCP.py

From Server:
 connectionSocket, addr = serverSocket.accept()
 sentence = connectionSocket.recv(1024).decode()

 file=open(sentence,"r")
 l=file.read(1024)

 connectionSocket.send(l.encode())
 print ('\nSent contents of ' + sentence)
 file.close()
 connectionSocket.close()
```

```
The server is ready to receive

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

Experiment No 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.



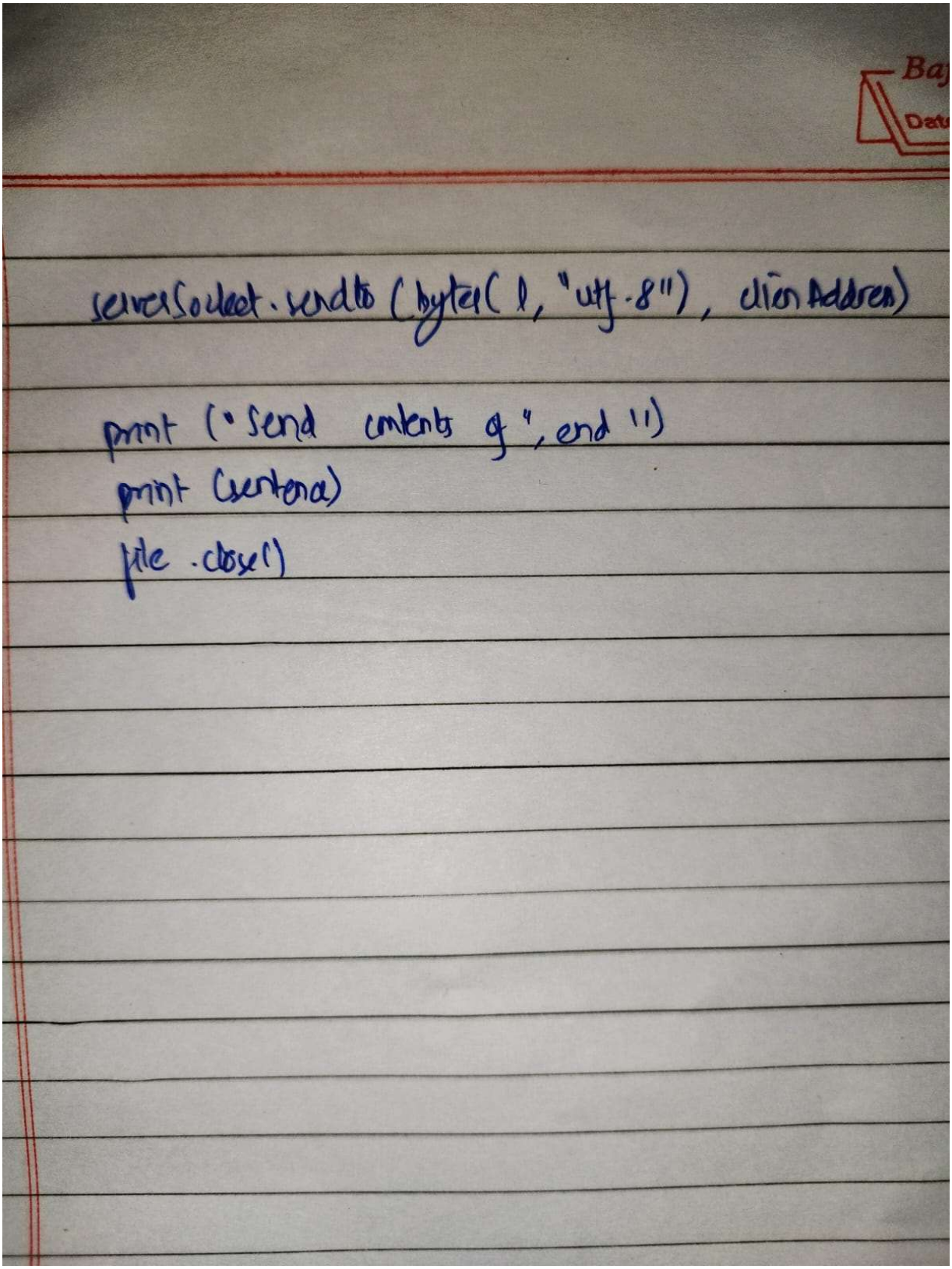
Client server program
UDP sockets

clientUDP.py

```
from socket import *
serverName = "127.0.0.1"
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)

sentence = input("File Name:")
clientSocket.sendto(bytes(sentence, "utf-8"), (serverName, serverPort))
fileContents, serverAddress = clientSocket.recv(2048)

print("Reply from server:")
print(fileContents.decode("utf-8"))
clientSocket.close()
clientSocket.close()
```

serverUDP.py

```
from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print("Server is ready to receive")
while(1):
    sentence, clientAddress = serverSocket.recvfrom(2048)
    sentence = sentence.decode("utf-8")
    file = open(sentence, "r")
    l = file.read()
```


serverSocket.sendto(bytes(l, "utf-8"), clientAddress)
print("Send contents of", end ")
print(contents)
file.close()

```
Enter file name: serverudp.py

Reply from Server:

from socket import *
serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(("127.0.0.1", serverPort))
print ("The server is ready to receive")
while 1:
 sentence, clientAddress = serverSocket.recvfrom(2048)
 sentence = sentence.decode("utf-8")
 file=open(sentence,"r")
 l=file.read(2048)

 serverSocket.sendto(bytes(l,"utf-8"),clientAddress)
 print ('\nSent contents of ', end = ' ')
 print (sentence)
 # for i in sentence:
 # print (str(i), end = '')
 file.close()
```

```
The server is ready to receive
```