INDEX

Ex No 1

LEARNING AND ANALYSING NETWORKING COMMANDS

Aim :

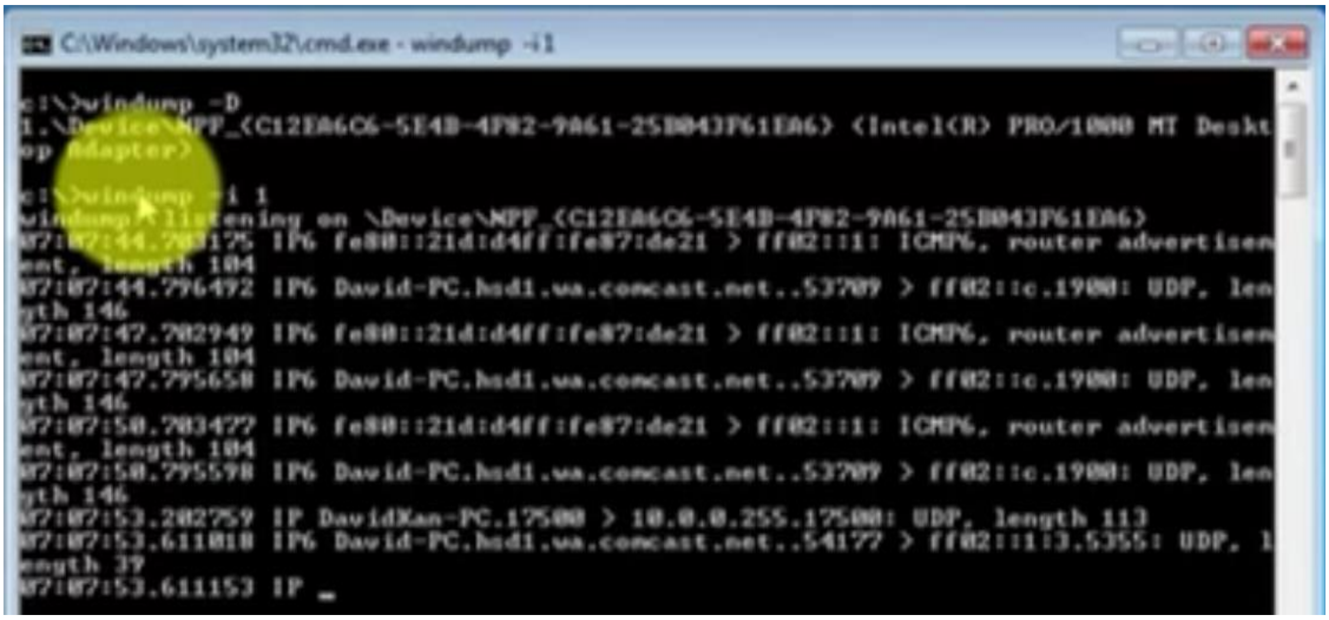
To Learn the use of commands like tcpdump, netstat, ifconfig, nslookup and traceroute. Capture ping and traceroute PDUs using a network protocol analyzer and examine.

NETWORING COMMANDS

1. **TCPDUMP**

Tcpdump is a common packet analyser that runs under the command line. It allows the user to display TCP/IP and other packets being transmitted or received over a network to which the computer is attached. Tcpdump works on most Unix-like operating systems: Linux, Solaris, FreeBSD, DragonFly BSD, NetBSD, OpenBSD, OpenWrt, macOS, HP-UX 11i, and AIX. In those systems, tcpdump uses the libpcap library to capture packets. The port of tcpdump for Windows is called WinDump; it uses WinPcap, the Windows port of libpcap.

Tcpdump was originally written in C Programming Language. Tcpdump prints the contents of network packets. It can read packets from a network interface card or from a previously created saved packet file. tcpdump can write packets to standard output or a file.



```
c:\>windump -D
1.\Device\NPF_{C12EA6C6-5E4B-4F82-9A61-25B043F61EA6} (Intel(R) PRO/1000 MT Deskt
op Adapter)

c:\>windump -i 1
windump: listening on \Device\NPF_{C12EA6C6-5E4B-4F82-9A61-25B043F61EA6}
07:07:44.703175 IP6 fe80::21d:d4ff:fe87:de21 > ff02::1: ICMP6, router advertisem
ent, length 104
07:07:44.796492 IP6 David-PC.hsd1.wa.comcast.net..53709 > ff02::c.1900: UDP, len
gth 146
07:07:47.702949 IP6 fe80::21d:d4ff:fe87:de21 > ff02::1: ICMP6, router advertisem
ent, length 104
07:07:47.795658 IP6 David-PC.hsd1.wa.comcast.net..53709 > ff02::c.1900: UDP, len
gth 146
07:07:50.703477 IP6 fe80::21d:d4ff:fe87:de21 > ff02::1: ICMP6, router advertisem
ent, length 104
07:07:50.795598 IP6 David-PC.hsd1.wa.comcast.net..53709 > ff02::c.1900: UDP, len
gth 146
07:07:53.202759 IP DavidXan-PC.17500 > 10.0.0.255.17500: UDP, length 113
07:07:53.611018 IP6 David-PC.hsd1.wa.comcast.net..54177 > ff02::1:3.5355: UDP, l
ength 39
07:07:53.611153 IP _
```

It is also possible to use tcpdump for the specific purpose of intercepting and displaying the communications of another user or computer. A user with the necessary privileges on a system acting as a router or gateway through which unencrypted traffic such as Telnet or HTTP passes can use tcpdump to view login IDs, passwords, the URLs and content of websites being viewed, or any other unencrypted information.

Example of available capture interfaces on a Linux system:

```
$ tcpdump -D
1.eth0 [Up, Running]
2.any (Pseudo-device that captures on all interfaces) [Up, Running]
3.lo [Up, Running, Loopback]
4.bluetooth-monitor (Bluetooth Linux Monitor)
5.nflog (Linux netfilter log (NFLOG) interface)
6.nfqueue (Linux netfilter queue (NFQUEUE) interface)
7.dbus-system (D-Bus system bus)
8.dbus-session (D-Bus session bus)
9.bluetooth0 (Bluetooth adapter number 0)
10.eth1
```

2. NETSTAT

Netstat (network statistics) is a command-line network utility that displays network connections for Transmission Control Protocol (both incoming and outgoing), routing tables, and a number of network interface (network interface controller or software-defined network interface) and network protocol statistics.

It is available on Unix-like operating systems including macOS, Linux, Solaris and BSD, and is available on IBM OS/2 and on Microsoft Windows NT-based operating systems including Windows XP, Windows Vista, Windows 7, Windows 8 and Windows 10. It is used for finding problems in the network and to determine the amount of traffic on the network as a performance measurement.

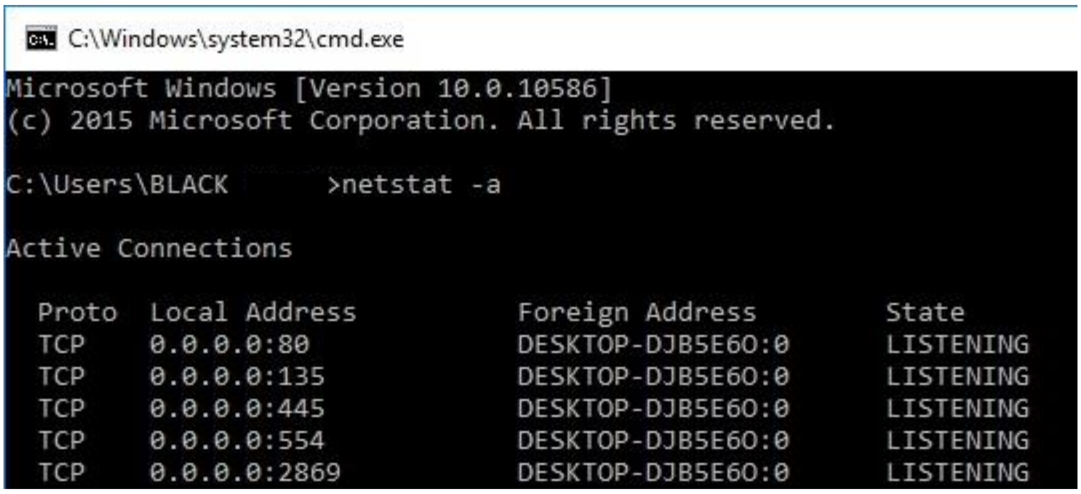
Check the commands as follow as on command prompt in Windows

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.
C:\Users\SYSTEM>netstat -a
```

The following netstat commands displays protocol statistics and current TCP/IP network connections.

Command	Description
netstat –a	Displays all connections, listening and non listening ports, sockets, protocols like tcp, udp etc
netstat –at	Shows only tcp connections (-au shows only udp connections)
netstat –ant	Shows all TCP connections with no dns resolution (show ip number instead).
netstat -atnp | grep ESTA	Displays all current Established TCP connections
netstat –al	Shows only listening sockets
netstat –b	Displays the executable involved in creating each connection or listening port. In some cases well-known executables host multiple independent components, and in these cases the sequence of components involved in creating the connection or listening port is displayed. In this case the

	executable name is in at the bottom, on top is the component it called, and so forth until TCP/IP was reached. Note that this option can be time-consuming and will fail unless you have sufficient permissions.
netstat -ct	Displays tcp connections continuously
netstat –e	Displays Ethernet statistics. This may be combined with the –s option.
netstat –f	Displays Fully Qualified Domain Names (FQDN) for foreign addresses.
netstat –g	Display multicast group membership information for IPv4 and IPv6
netstat –i	Displays a table of all network interfaces. Add -e to get output similar to ifconfig
netstat –lntu	Display all services listening for tcp and udp, all free open ports on the local machine
netstat –n	Displays addresses and port numbers in numerical form.
netstat –o	Displays the owning process ID associated with each connection.
netstat –p protocol name	Shows connections for the protocol specified by proto; proto may be any of: TCP, UDP, TCPv6, or UDPv6. If used with the –s option to display per-protocol statistics, proto may be any of: IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, or UDPv6.
netstat -q	Displays all connections, listening ports, and bound non listening TCP ports. Bound non listening ports may or may not be associated with an active connection.
netstat –r	Displays the routing table.
netstat –s	Displays per-protocol statistics. By default, statistics are shown for IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, and UDPv6; the -p option may be used to specify a subset of the default.
netstat –t	Displays the current connection offload state.
netstat –x	Displays Network Direct connections, listeners, and shared endpoints.
netstat –y	Displays the TCP connection template for all connections. Cannot be combined with the other options.
netstat –interval	Redisplays selected statistics, pausing interval seconds between each display. Press CTRL+C to stop redisplaying statistics. If omitted, netstat will print the current configuration information once.



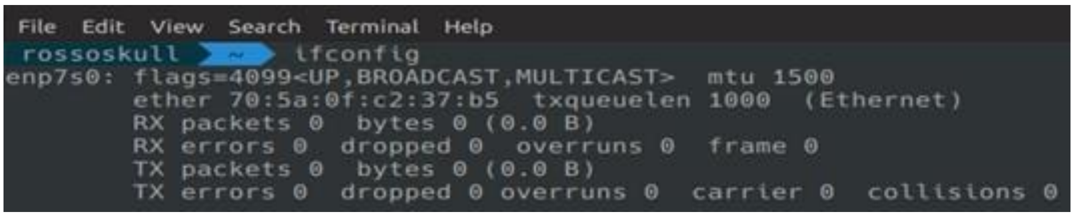
```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK      >netstat -a

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:80             DESKTOP-DJB5E6O:0      LISTENING
  TCP    0.0.0.0:135            DESKTOP-DJB5E6O:0      LISTENING
  TCP    0.0.0.0:445            DESKTOP-DJB5E6O:0      LISTENING
  TCP    0.0.0.0:554            DESKTOP-DJB5E6O:0      LISTENING
  TCP    0.0.0.0:2869           DESKTOP-DJB5E6O:0      LISTENING
```

3. IFCONFIG

ifconfig is a system administration utility in Unix-like operating systems for network interface configuration. The utility is a command-line interface tool and is also used in the system startup scripts of many operating systems. It has features for configuring, controlling, and querying TCP/IP network interface parameters.

Syntax: **ifconfig [...OPTIONS] [INTERFACE]**

ifconfig -a	Display all the interfaces available, even if they are down.
ifconfig -s	Display a short list, instead of details.
ifconfig -v	Run the command in verbose mode – log more details about execution.
ifconfig interface up	Activate the driver for the given interface.
ifconfig interface down	Deactivate the driver for the given interface.
ifconfig interface add addr/prefixlen	Add an IPv6 address to an interface.
ifconfig interface del addr/prefixlen	Remove an IPv6 address to an interface.
ifconfig interface [-]arp	Enable/disable the use of ARP protocol on an interface.
ifconfig interface [-]promisc	Enable/disable the promiscuous mode on an interface. If it is selected, all the packets on the network will be received by the interface.
ifconfig interface [-]allmulti	Enable/disable all-multicast mode for an interface. If it is selected, all the multicast packets will be received by the interface.
ifconfig interface [-]allmulti	Set the Maximum Transfer Unit(MTU).
ifconfig --help	Display help related to ifconfig command.



```
 rossoskull  ~  ifconfig
enp7s0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        ether 70:5a:0f:c2:37:b5  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

4. NSLOOKUP

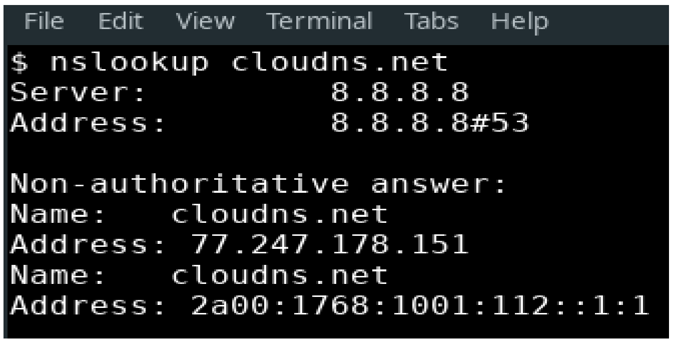
Nslookup is a network administration command-line tool available in many computer operating systems for querying the Domain Name System (DNS) to obtain domain name or IP address mapping, or other DNS records. The name "nslookup" means "name server lookup". This tool can be used to check DNS records propagation using different servers, to confirm proper DNS resolution, and perform other troubleshooting steps and can use via the command prompt as follow as.

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.
C:\Users\SYSTEM>nslookup example.com
```

To find the A record of a domain	- nslookup example.com
To check the NS records of a domain	- nslookup -type=ns example.com
To query the SOA record of a domain	- nslookup -type=soa example.com
To find the MX records responsible for the email exchange	- nslookup -query=mx example.com
To find all of the available DNS records of a domain	- nslookup -type=any example.com
To check the using of a specific DNS Server	- nslookup example.com ns1.nsexample.com
To check the Reverse DNS Lookup	- nslookup 10.20.30.40
To change the port number for the connection	- nslookup -port=56 example.com
To change the timeout interval for a reply	- nslookup -timeout=20 example.com
To enable debug mode	- nslookup -debug example.com

Debug mode provides important and detailed information both for the question and for the received answer.

Notes: Authoritative answer – This is the answer that originates from the DNS Server which has the information about the zone file. Non-authoritative answer – When a nameserver is not in the list for the domain you did a lookup on. Different port – By default, the DNS servers use port 53.



```
$ nslookup cloudns.net
Server:         8.8.8.8
Address:        8.8.8.8#53

Non-authoritative answer:
Name:   cloudns.net
Address: 77.247.178.151
Name:   cloudns.net
Address: 2a00:1768:1001:112::1:1
```

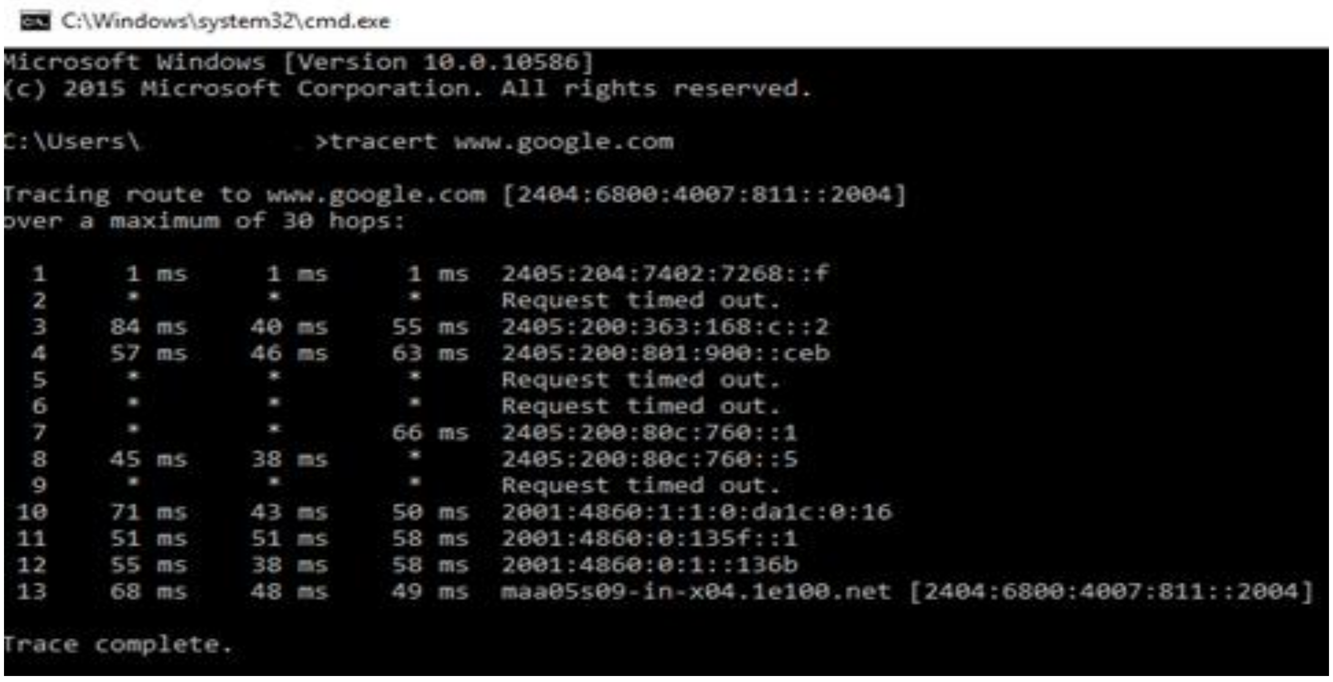
5. TRACEROUTE

A traceroute is a function which traces the path from one network to another. Traceroute and tracert are computer network diagnostic commands for displaying the route (path) and measuring transit delays of packets across an Internet Protocol (IP) network. The history of the route is recorded as the round-trip times of the packets received from each successive host (remote node) in the route (path); the sum of the mean times in each hop is a measure of the total time spent to establish the connection. Traceroute proceeds unless all (three) sent packets are lost more than twice; then the connection is lost and the route cannot be evaluated. Ping, on the other hand, only computes the final round-trip times from the destination point. For Internet Protocol Version 6 (IPv6) the tool sometimes has the name traceroute6 or tracert6.

```
C:\Users\BLACK BILLA>tracert /?
Usage: tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout]
               [-R] [-S srcaddr] [-4] [-6] target_name
Options:
```

-d	Do not resolve addresses to hostnames.
-h maximum_hops	Maximum number of hops to search for target.
-j host-list	Loose source route along host-list (IPv4-only).
-w timeout	Wait timeout milliseconds for each reply.
-R	Trace round-trip path (IPv6-only).
-S srcaddr	Source address to use (IPv6-only).
-4	Force using IPv4.
-6	Force using IPv6

Example : tracert -4 www.google.com



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\           >tracert www.google.com

Tracing route to www.google.com [2404:6800:4007:811::2004]
over a maximum of 30 hops:

  1     1 ms     1 ms     1 ms  2405:204:7402:7268::f
  2     *        *        *     Request timed out.
  3    84 ms    40 ms    55 ms  2405:200:363:168:c::2
  4    57 ms    46 ms    63 ms  2405:200:801:900::ceb
  5     *        *        *     Request timed out.
  6     *        *        *     Request timed out.
  7     *        *       66 ms  2405:200:80c:760::1
  8    45 ms    38 ms     *     2405:200:80c:760::5
  9     *        *        *     Request timed out.
 10    71 ms    43 ms    50 ms  2001:4860:1:1:0:da1c:0:16
 11    51 ms    51 ms    58 ms  2001:4860:0:135f::1
 12    55 ms    38 ms    58 ms  2001:4860:0:1::136b
 13    68 ms    48 ms    49 ms  maa05s09-in-x04.1e100.net [2404:6800:4007:811::2004]

Trace complete.
```

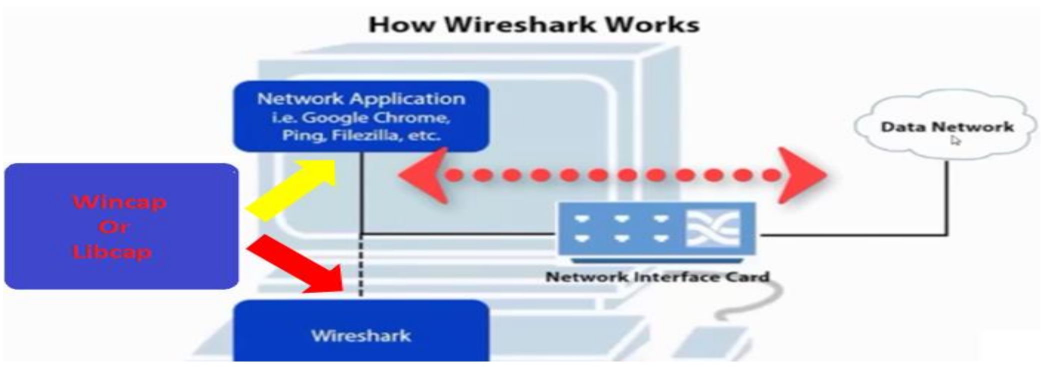
WIRESHARK – THE NETWORK PROTOCOL ANALYZER

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Wireshark is cross-platform, using the Qt widget toolkit in current releases to implement its user interface, and using pcap to capture packets. Wireshark runs on Linux, macOS, BSD, Solaris, some other Unix-like operating systems, and Microsoft Windows. There is also a terminal-based (non-GUI) version called TShark.

Wireshark is very similar to tcpdump, but has a graphical front-end, plus some integrated sorting and filtering options. Wireshark has wincap driver for windows platform and libcap driver for linux platform to capture packets while the client sends the request, get response to and from server.

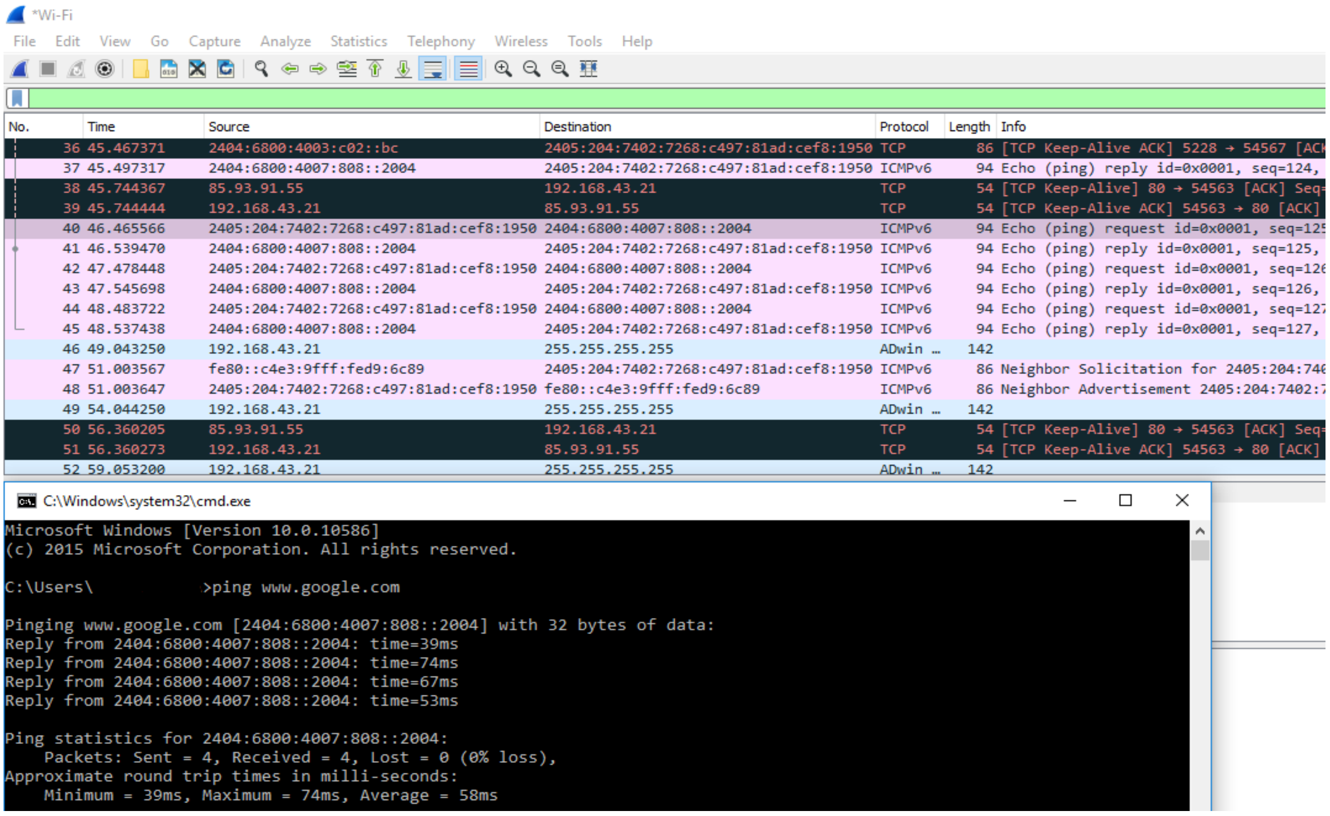
Wireshark lets the user put network interface controllers into promiscuous mode (if supported by the network interface controller), so they can see all the traffic visible on that interface including unicast traffic not sent to that network interface controller's MAC address. However, when capturing with a packet analyzer in promiscuous mode on a port on a network switch, not all traffic through the switch is necessarily sent to the port where the capture is done, so capturing in promiscuous mode is not necessarily sufficient to see all network traffic. Port mirroring or various network taps extend capture to any point on the network. Simple passive taps are extremely resistant to tampering.

If a remote machine captures packets and sends the captured packets to a machine running Wireshark using the TZSP protocol or the protocol used by OmniPeek, Wireshark dissects those packets, so it can analyze packets captured on a remote machine at the time that they are captured.



STEPS :

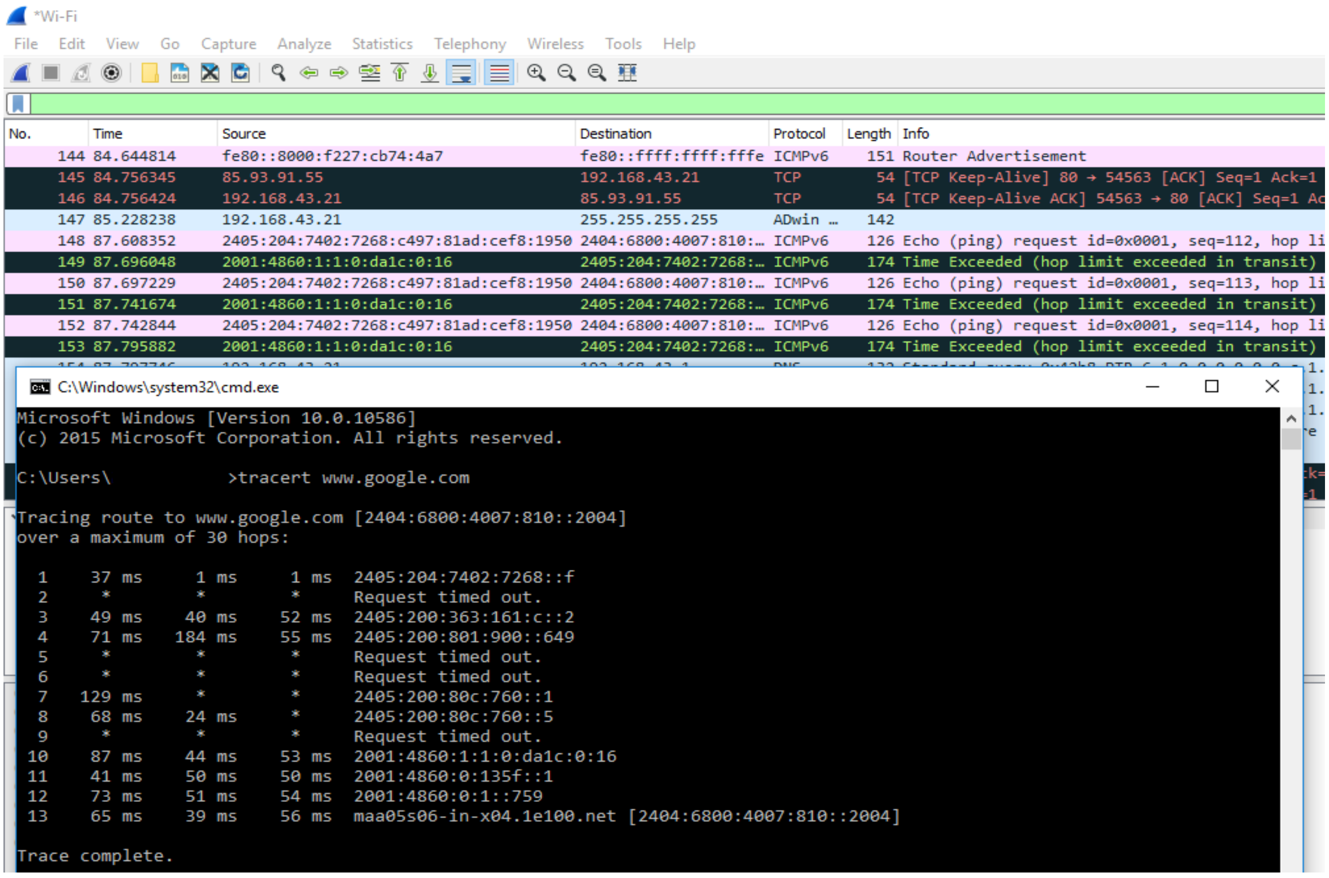
1. Open Wireshark, check the graph representation of installed network controller interfaces
2. Set the preferences form edit menu to display the required field while capturing the packets.
3. Select “Option” option from capture menu, select current network interfaces from input tab, set required preferences from output and options tab in Wireshark-Capture Interfaces window, make sure enable promiscuous option is checked and click start.
4. Now the packet capturing starts and open windows command prompt type > “ping ip address” response will appear in command prompt, simultaneously watch the packet capturing in Wireshark capturing window.
5. Similarly check the traceroute command in Wireshark.


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
No.  Time       Source                                  Destination                             Protocol  Length Info
 36 45.467371   2404:6800:4003:c02::bc                  2405:204:7402:7268:c497:81ad:cef8:1950  TCP        86 [TCP Keep-Alive ACK] 5228 → 54567 [ACK
 37 45.497317   2404:6800:4007:808::2004                2405:204:7402:7268:c497:81ad:cef8:1950  ICMPv6     94 Echo (ping) reply id=0x0001, seq=124,
 38 45.744367   85.93.91.55                             192.168.43.21                           TCP        54 [TCP Keep-Alive] 80 → 54563 [ACK] Seq=
 39 45.744444   192.168.43.21                           85.93.91.55                             TCP        54 [TCP Keep-Alive ACK] 54563 → 80 [ACK]
 40 46.465566   2405:204:7402:7268:c497:81ad:cef8:1950  2404:6800:4007:808::2004                ICMPv6     94 Echo (ping) request id=0x0001, seq=125
 41 46.539470   2404:6800:4007:808::2004                2405:204:7402:7268:c497:81ad:cef8:1950  ICMPv6     94 Echo (ping) reply id=0x0001, seq=125,
 42 47.478448   2405:204:7402:7268:c497:81ad:cef8:1950  2404:6800:4007:808::2004                ICMPv6     94 Echo (ping) request id=0x0001, seq=126
 43 47.545698   2404:6800:4007:808::2004                2405:204:7402:7268:c497:81ad:cef8:1950  ICMPv6     94 Echo (ping) reply id=0x0001, seq=126,
 44 48.483722   2405:204:7402:7268:c497:81ad:cef8:1950  2404:6800:4007:808::2004                ICMPv6     94 Echo (ping) request id=0x0001, seq=127
 45 48.537438   2404:6800:4007:808::2004                2405:204:7402:7268:c497:81ad:cef8:1950  ICMPv6     94 Echo (ping) reply id=0x0001, seq=127,
 46 49.043250   192.168.43.21                           255.255.255.255                         ADwin …   142
 47 51.003567   fe80::c4e3:9fff:fed9:6c89               2405:204:7402:7268:c497:81ad:cef8:1950  ICMPv6     86 Neighbor Solicitation for 2405:204:740
 48 51.003647   2405:204:7402:7268:c497:81ad:cef8:1950  fe80::c4e3:9fff:fed9:6c89               ICMPv6     86 Neighbor Advertisement 2405:204:7402:7
 49 54.044250   192.168.43.21                           255.255.255.255                         ADwin …   142
 50 56.360205   85.93.91.55                             192.168.43.21                           TCP        54 [TCP Keep-Alive] 80 → 54563 [ACK] Seq=
 51 56.360273   192.168.43.21                           85.93.91.55                             TCP        54 [TCP Keep-Alive ACK] 54563 → 80 [ACK]
 52 59.053200   192.168.43.21                           255.255.255.255                         ADwin …   142
```

C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\             >ping www.google.com

Pinging www.google.com [2404:6800:4007:808::2004] with 32 bytes of data:
Reply from 2404:6800:4007:808::2004: time=39ms
Reply from 2404:6800:4007:808::2004: time=74ms
Reply from 2404:6800:4007:808::2004: time=67ms
Reply from 2404:6800:4007:808::2004: time=53ms

Ping statistics for 2404:6800:4007:808::2004:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 39ms, Maximum = 74ms, Average = 58ms
```


*Wi-Fi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

```
No.  Time        Source                                  Destination              Protocol  Length Info
 144 84.644814   fe80::8000:f227:cb74:4a7                fe80::ffff:ffff:fffe     ICMPv6    151 Router Advertisement
 145 84.756345   85.93.91.55                             192.168.43.21            TCP        54 [TCP Keep-Alive] 80 → 54563 [ACK] Seq=1 Ack=1
 146 84.756424   192.168.43.21                           85.93.91.55              TCP        54 [TCP Keep-Alive ACK] 54563 → 80 [ACK] Seq=1 Ac
 147 85.228238   192.168.43.21                           255.255.255.255          ADwin …   142
 148 87.608352   2405:204:7402:7268:c497:81ad:cef8:1950  2404:6800:4007:810:…     ICMPv6    126 Echo (ping) request id=0x0001, seq=112, hop li
 149 87.696048   2001:4860:1:1:0:da1c:0:16               2405:204:7402:7268:…     ICMPv6    174 Time Exceeded (hop limit exceeded in transit)
 150 87.697229   2405:204:7402:7268:c497:81ad:cef8:1950  2404:6800:4007:810:…     ICMPv6    126 Echo (ping) request id=0x0001, seq=113, hop li
 151 87.741674   2001:4860:1:1:0:da1c:0:16               2405:204:7402:7268:…     ICMPv6    174 Time Exceeded (hop limit exceeded in transit)
 152 87.742844   2405:204:7402:7268:c497:81ad:cef8:1950  2404:6800:4007:810:…     ICMPv6    126 Echo (ping) request id=0x0001, seq=114, hop li
 153 87.795882   2001:4860:1:1:0:da1c:0:16               2405:204:7402:7268:…     ICMPv6    174 Time Exceeded (hop limit exceeded in transit)
```

C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\             >tracert www.google.com

Tracing route to www.google.com [2404:6800:4007:810::2004]
over a maximum of 30 hops:

  1    37 ms     1 ms     1 ms  2405:204:7402:7268::f
  2     *        *        *     Request timed out.
  3    49 ms    40 ms    52 ms  2405:200:363:161:c::2
  4    71 ms   184 ms    55 ms  2405:200:801:900::649
  5     *        *        *     Request timed out.
  6     *        *        *     Request timed out.
  7   129 ms     *        *     2405:200:80c:760::1
  8    68 ms    24 ms     *     2405:200:80c:760::5
  9     *        *        *     Request timed out.
 10    87 ms    44 ms    53 ms  2001:4860:1:1:0:da1c:0:16
 11    41 ms    50 ms    50 ms  2001:4860:0:135f::1
 12    73 ms    51 ms    54 ms  2001:4860:0:1::759
 13    65 ms    39 ms    56 ms  maa05s06-in-x04.1e100.net [2404:6800:4007:810::2004]

Trace complete.
```

RESULT :

The learning, analysing, capturing and examining of networking commands has been done successfully.

Ex.No : 2

DATE :

WRITE A HTTP WEB CLIENT PROGRAM TO DOWNLOAD A WEB PAGE USING TCP SOCKETS

AIM:

To write a java program for socket for HTTP for web page upload and download .

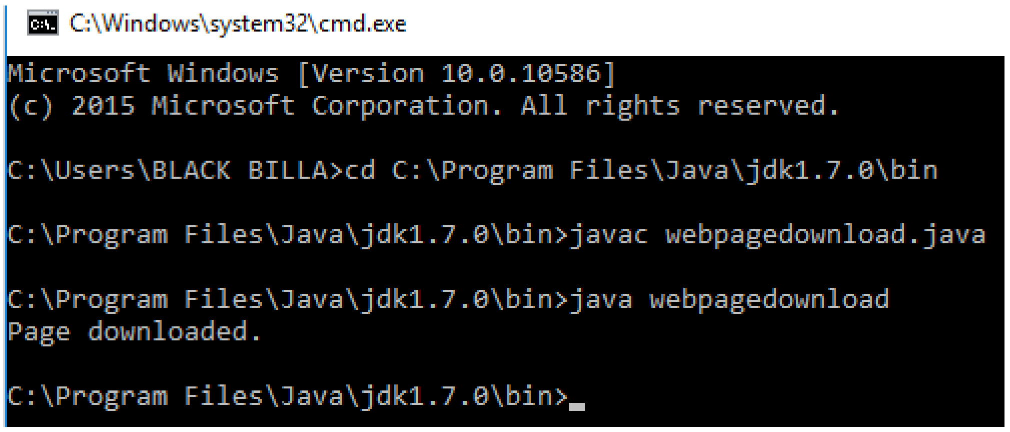
ALGORITHM:

1. Start.
2. Create socket and establish the connection with the server.
3. Read the image to be uploaded from the disk
4. Send the image read to the server
5. Terminate the connection
6. Stop.

PROGRAM

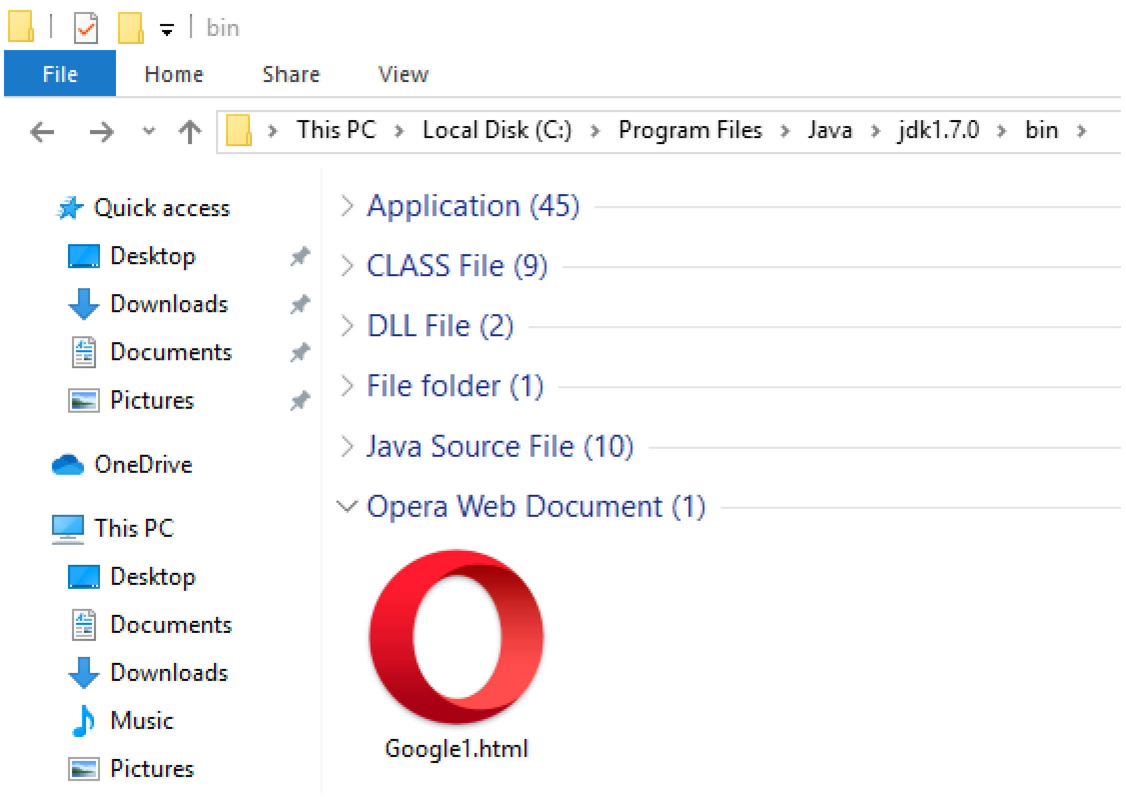
```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.URL;
public class Download
{
  public static void main(String args[]) throws IOException
{
    download("http://www.google.com");
  }
  public static void download(String urlString) throws IOException
{
    URL url = new URL(urlString);
    try(
      BufferedReader reader =  new BufferedReader(new
InputStreamReader(url.openStream()));
      BufferedWriter writer = new BufferedWriter(new FileWriter("Google.html"));
    )
{
      String line;
      while ((line = reader.readLine()) != null)
        {
        writer.write(line);
        }
      System.out.println("Page downloaded.");
    }
  }
}
```

OUTPUT :



C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac webpagedownload.java

C:\Program Files\Java\jdk1.7.0\bin>java webpagedownload
Page downloaded.

C:\Program Files\Java\jdk1.7.0\bin>_
```



RESULT :

Thus the program for creating sockets for HTTP net page to download for implemented .

Ex.No: 3(a) **APPLICATIONS USING TCP SOCKETS LIKE: ECHO CLIENT AND ECHO SERVER,**

DATE :

AIM

To write a java program for application using TCP Sockets Links

ALGORITHM

Client

1. Start
2. Create the TCP socket
3. Establish connection with the server
4. Get the message to be echoed from the user
5. Send the message to the server
6. Receive the message echoed by the server
7. Display the message received from the server
8. Terminate the connection
9. Stop

Server

1. Start
2. Create TCP socket, make it a listening socket
3. Accept the connection request sent by the client for connection establishment
4. Receive the message sent by the client
5. Display the received message
6. Send the received message to the client from which it receives
7. Close the connection when client initiates termination and server becomes a listening server,waiting for clients.
8. Stop.

PROGRAM :

ECHOCLIENT :-

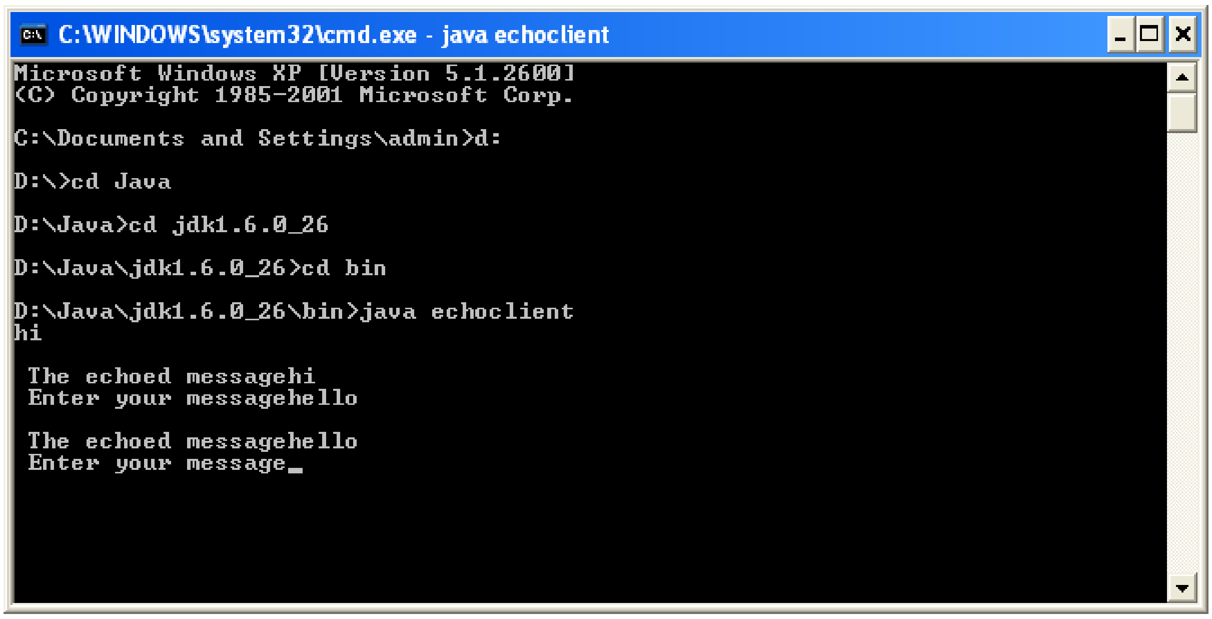
```
import java.io.*;
import java.net.*;
import java.util.*;
public class echoclient
{
public static void main(String args[])throws Exception
{
Socket c=null;
DataInputStream usr_inp=null;
DataInputStream din=new DataInputStream(System.in);
DataOutputStream dout=null;
try
{
c=new Socket("127.0.0.1",5678);
usr_inp=new DataInputStream(c.getInputStream());
dout=new DataOutputStream(c.getOutputStream());
}
catch(IOException e)
{
}
if(c!=null || usr_inp!=null || dout!=null)
{
String unip;
while((unip=din.readLine())!=null)
{
```

```
dout.writeBytes(""+unip);
dout.writeBytes("\n");
System.out.print("\n The echoed message");
System.out.print(usr_inp.readLine());
System.out.print("\n Enter your message");
}
System.exit(0);
}
din.close();
usr_inp.close();
c.close();
}
}
```

ECHO SERVER :-

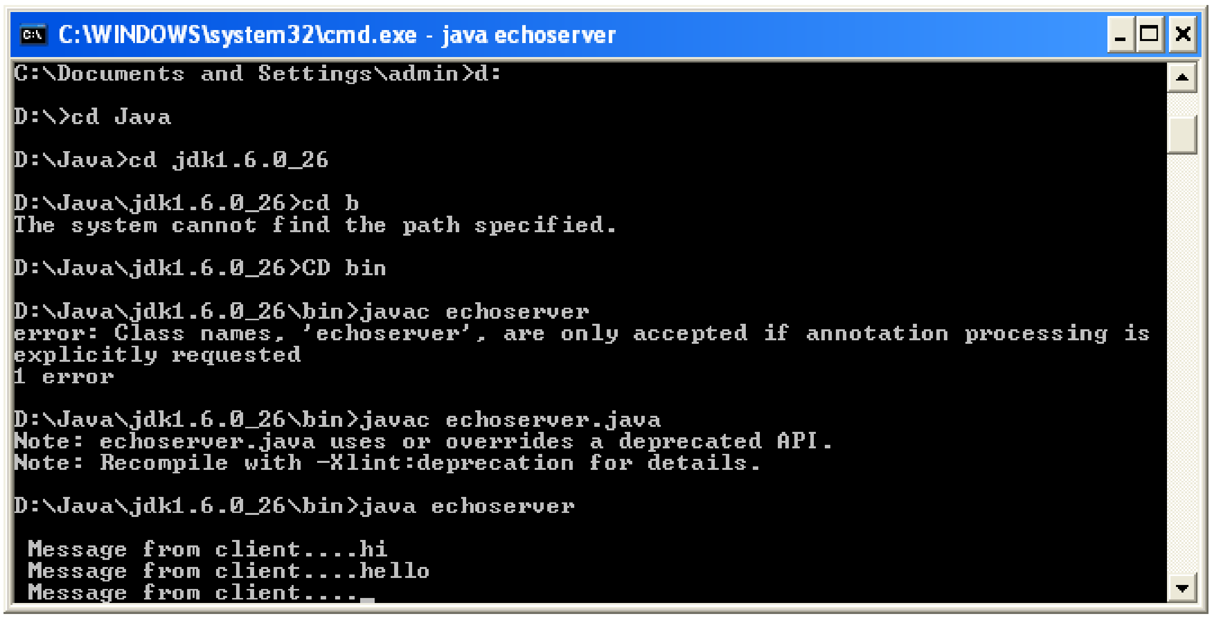
```
import java.io.*;
import java.net.*;
public class echoserver
{
public static void main(String[]args)throws Exception
{
ServerSocket m=null;
Socket c=null;
DataInputStream usr_inp=null;
DataInputStream din=new DataInputStream(System.in);
DataOutputStream dout=null;
try
{
m=new ServerSocket(5678);
c=m.accept();
usr_inp=new DataInputStream(c.getInputStream());
dout=new DataOutputStream(c.getOutputStream());
}
catch(IOException e)
{
}
if(c!=null || usr_inp!=null)
{
String unip;
while(true)
{
System.out.print("\n Message from client....");
String m1=(usr_inp.readLine());
System.out.print(m1);
dout.writeBytes(""+m1);
dout.writeBytes("\n");
}
}
dout.close();
usr_inp.close();
c.close();
}
}
```

CLIENT AND SERVER OUTPUT :



C:\WINDOWS\system32\cmd.exe - java echoclient

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\admin>d:

D:\>cd Java

D:\Java>cd jdk1.6.0_26

D:\Java\jdk1.6.0_26>cd bin

D:\Java\jdk1.6.0_26\bin>java echoclient
hi

 The echoed messagehi
 Enter your messagehello

 The echoed messagehello
 Enter your message_
```



C:\WINDOWS\system32\cmd.exe - java echoserver

```
C:\Documents and Settings\admin>d:

D:\>cd Java

D:\Java>cd jdk1.6.0_26

D:\Java\jdk1.6.0_26>cd b
The system cannot find the path specified.

D:\Java\jdk1.6.0_26>CD bin

D:\Java\jdk1.6.0_26\bin>javac echoserver
error: Class names, 'echoserver', are only accepted if annotation processing is
explicitly requested
1 error

D:\Java\jdk1.6.0_26\bin>javac echoserver.java
Note: echoserver.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

D:\Java\jdk1.6.0_26\bin>java echoserver

 Message from client....hi
 Message from client....hello
 Message from client...._
```

RESULT:

Thus the java program to concurrently using TCP sockets was excited successfully

Ex.No: 3(b) **CHAT**

DATE :

AIM:

To implement a chat application using TCP sockets

ALGORITHM

Client

1. Start
2. Create the UDP datagram socket
3. Get the request message to be sent from the user
4. Send the request message to the server
5. If the request message is “END” go to step 10
6. Wait for the reply message from the server
7. Receive the reply message sent by the server
8. Display the reply message received from the server
9. Repeat the steps from 3 to 8
10. Stop

Server

1. Start
2. Create UDP datagram socket, make it a listening socket
3. Receive the request message sent by the client
4. If the received message is “END” go to step 10
5. Retrieve the client’s IP address from the request message received
6. Display the received message
7. Get the reply message from the user
8. Send the reply message to the client
9. Repeat the steps from 3 to 8.
10. Stop.

PROGRAM:

Chat Server :

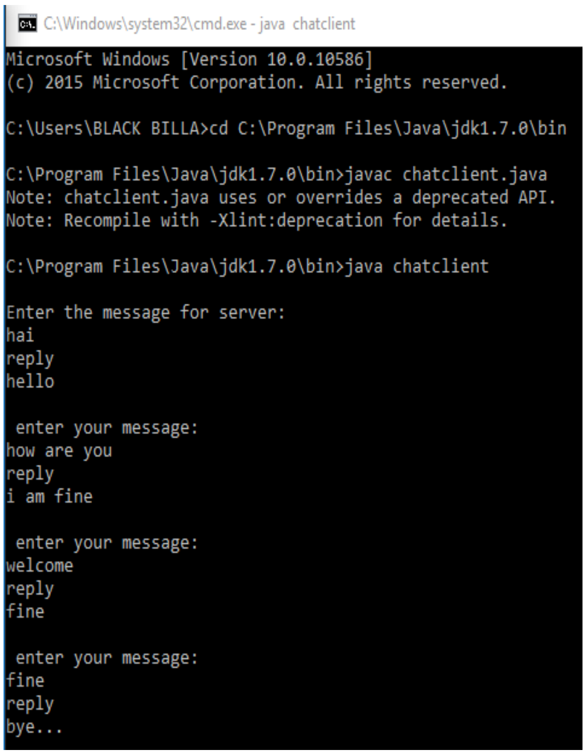
```
import java.io.*;
import java.net.*;
public class chatserver
{
public static void main(String args[])throws Exception
{
ServerSocket m=null;
Socket c=null;
DataInputStream usr_inp=null;
DataInputStream din=new DataInputStream(System.in);
DataOutputStream dout=null;
try
{
m=new ServerSocket(1234);
c=m.accept();
usr_inp=new DataInputStream(c.getInputStream());
dout=new DataOutputStream(c.getOutputStream());
}
catch(IOException e)
{
}
if(c!=null || usr_inp!=null)
{
String unip;
while(true)
{
System.out.println("\n message from client");
String ml=usr_inp.readLine();
System.out.println(ml);
System.out.println("\n enter your message:");
unip=din.readLine();
dout.writeBytes(""+unip);
dout.writeBytes("\n");
}
}
dout.close();
usr_inp.close();
c.close();
}
}
```

Chat Client :

```
import java.io.*;
import java.net.*;
public class chatclient
{
public static void main(String args[])throws Exception
{
Socket c=null;
DataInputStream usr_inp=null;
DataInputStream din=new DataInputStream(System.in);
DataOutputStream dout=null;
try
{
c=new Socket("127.0.0.1",1234);
usr_inp=new DataInputStream(c.getInputStream());
dout=new DataOutputStream(c.getOutputStream());
}
catch(IOException e)
{
}
if(c!=null || usr_inp!=null || dout!=null)
{
String unip;
System.out.println("\nEnter the message for server:");
while((unip=din.readLine())!=null)
{
dout.writeBytes(""+unip);
dout.writeBytes("\n");
System.out.println("reply");
System.out.println(usr_inp.readLine());
System.out.println("\n enter your message:");
}
System.exit(0);
}
din.close();
usr_inp.close();
c.close();
}
}
```

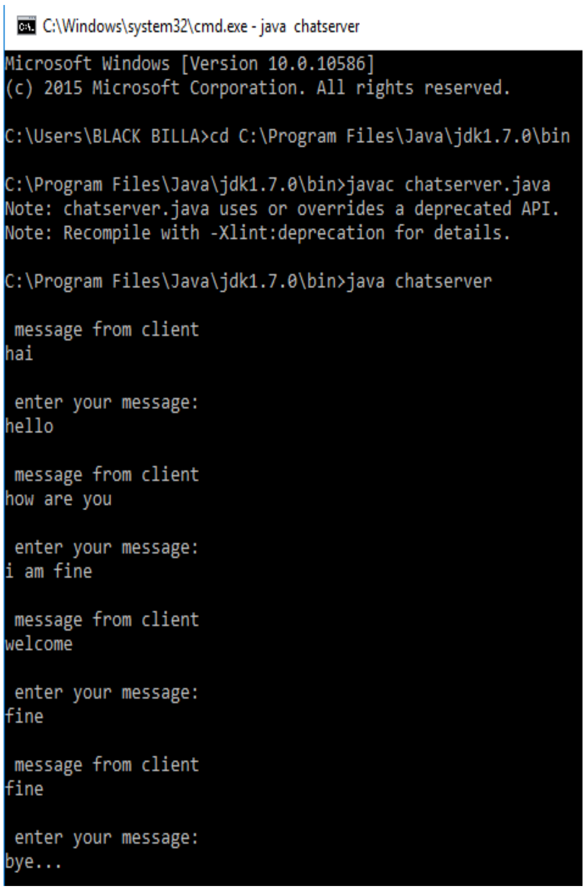
OUTPUT:

CLIENT OUTPUT :



```
C:\Windows\system32\cmd.exe - java  chatclient
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac chatclient.java
Note: chatclient.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Program Files\Java\jdk1.7.0\bin>java chatclient

Enter the message for server:
hai
reply
hello

 enter your message:
how are you
reply
i am fine

 enter your message:
welcome
reply
fine

 enter your message:
fine
reply
bye...
```

SERVER OUTPUT:



```
C:\Windows\system32\cmd.exe - java  chatserver
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac chatserver.java
Note: chatserver.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Program Files\Java\jdk1.7.0\bin>java chatserver

 message from client
hai

 enter your message:
hello

 message from client
how are you

 enter your message:
i am fine

 message from client
welcome

 enter your message:
fine

 message from client
fine

 enter your message:
bye...
```

RESULT:

Chat application using TCP Sockets Implemated Successfully.

Ex.No: 3(c) **FILE TRANSFER**

DATE :

AIM:

To write a java program for file transfer using TCP Sockets.

Algorithm

Server

1. Import java packages and create class file server.
2. Create a new server socket and bind it to the port.
3. Accept the client connection
4. Get the file name and stored into the BufferedReader.
5. Create a new object class file and realine.
6. If file is exists then FileReader read the content until EOF is reached.
7. Stop the program.

Client

1. Import java packages and create class file server.
2. Create a new server socket and bind it to the port.
3. Now connection is established.
4. The object of a BufferReader class is used for storing data content which has been retrievedfrom socket object.
5. The connection is closed.
6. Stop the program.

PROGRAM:

Server Ftp :

```
import java.io.*;
import java.net.*;
import java.util.*;
class Serverftp
{
public static void main(String args[])
{
try
{
ServerSocket obj=new ServerSocket(137);
while(true)
{
Socket obj1=obj.accept();
DataInputStream din=new DataInputStream(obj1.getInputStream());
DataOutputStream dout=new DataOutputStream(obj1.getOutputStream());
System.out.println("Enter the file name:");
String str=din.readLine();
FileReader f=new FileReader(str);
BufferedReader b=new BufferedReader(f);
String s;
while((s=b.readLine())!=null)
{
System.out.println(s);
dout.writeBytes(s+"\n");
}
f.close();
dout.writeBytes("- I\n");
}
}
catch(IOException e)
{
```

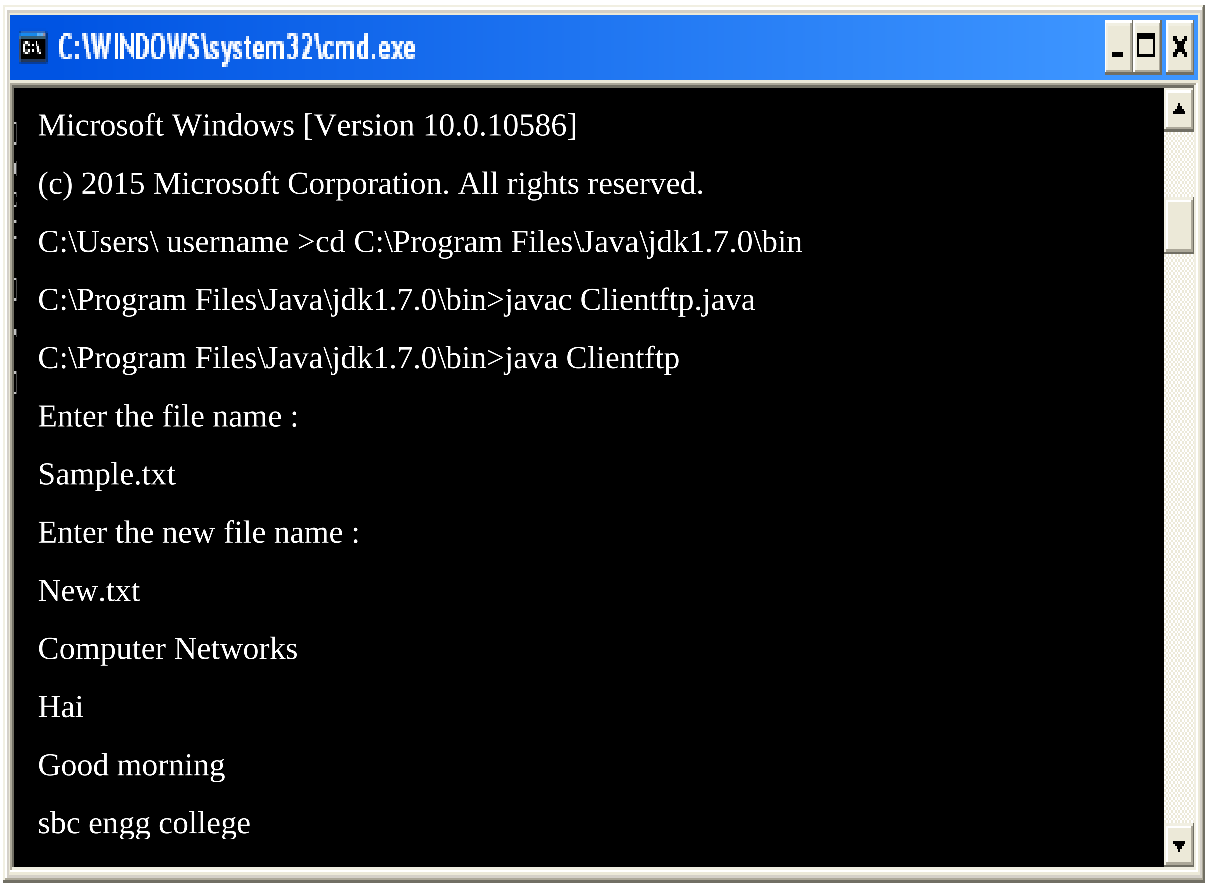
```
System.out.println(e);
}
}
}
```

Client Ftp :

```
import java.io.*;
import java.net.*;
 import java.util.*;
 class Clientftp
{
public static void main(String args[])
{
try
{
BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
Socket clsct=new Socket("127.0.0.1",137);
DataInputStream din=new DataInputStream(clsct.getInputStream());
DataOutputStream dout=new DataOutputStream(clsct.getOutputStream());
System.out.println("Enter the file name:");
String str=din.readLine();
dout.writeBytes(str+'\n');
System.out.println("Enter the file name: ");
String str2=din.readLine();
String str1,ss;
FileWriter f=new FileWriter(str2);
char buffer[];
while(true)
{
str1=din.readLine();
if(str1.equals("-1"))
break;
System.out.println(str1);
buffer=new char[str1.length()];
str1.getChars(0,str1.length(),buffer,0);
f.write(buffer);
}
f.close();
clsct.close();
}
catch(IOException e)
{
System.out.println(e);
}
}
}
```

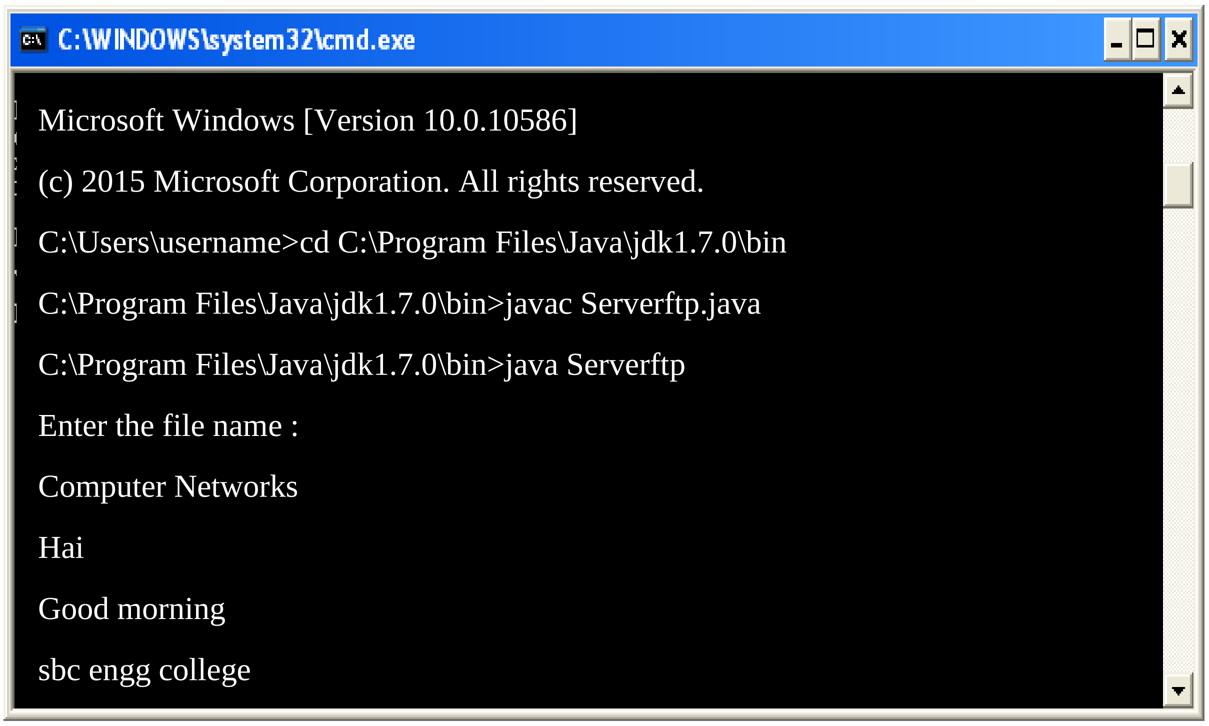
OUTPUT:

CLIENT OUTPUT :



```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.
C:\Users\ username >cd C:\Program Files\Java\jdk1.7.0\bin
C:\Program Files\Java\jdk1.7.0\bin>javac Clientftp.java
C:\Program Files\Java\jdk1.7.0\bin>java Clientftp
Enter the file name :
Sample.txt
Enter the new file name :
New.txt
Computer Networks
Hai
Good morning
sbc engg college
```

SERVER OUTPUT:



```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.
C:\Users\username>cd C:\Program Files\Java\jdk1.7.0\bin
C:\Program Files\Java\jdk1.7.0\bin>javac Serverftp.java
C:\Program Files\Java\jdk1.7.0\bin>java Serverftp
Enter the file name :
Computer Networks
Hai
Good morning
sbc engg college
```

RESULT :

Thus the java program file transfer application using TCP sockets was executed succesfully .

Ex.No : 4 **Simulation of DNS using UDP Sockets**

DATE :

AIM:

Write java program simulation of DNS using UDP sockets

ALGORITHM

Server

1. Start
2. Create UDP datagram socket
3. Create a table that maps host name and IP address
4. Receive the host name from the client
5. Retrieve the client's IP address from the received datagram
6. Get the IP address mapped for the host name from the table.
7. Display the host name and corresponding IP address
8. Send the IP address for the requested host name to the client
9. Stop.

Client

1. Start
2. Create UDP datagram socket.
3. Get the host name from the client
4. Send the host name to the server
5. Wait for the reply from the server
6. Receive the reply datagram and read the IP address for the requested host name
7. Display the IP address.
8. Stop.

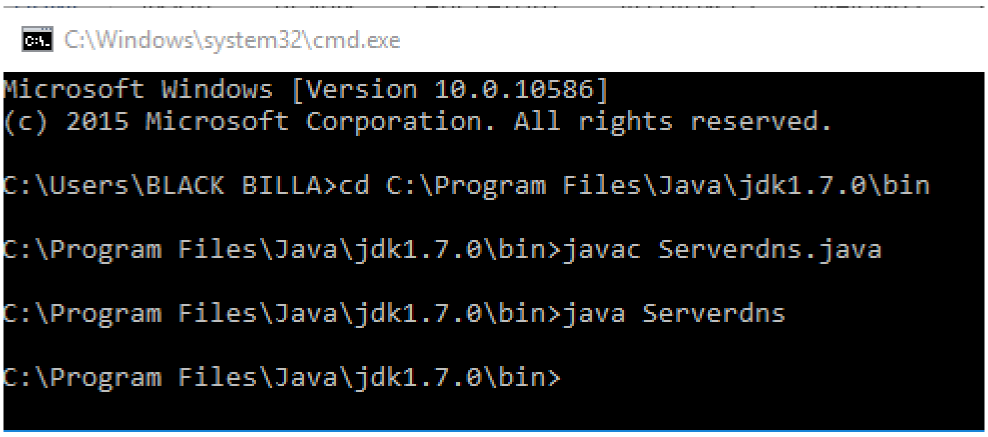
SERVER PROGAM

```
import java.io.*;
import java.net.*;
import java.util.*;
class Serverdns12
{
public static void main(String args[])
{
try
{
DatagramSocket server=new DatagramSocket(1309);
while(true)
{
byte[] sendbyte=new byte[1024];
byte[] receivebyte=new byte[1024];
DatagramPacket receiver=new DatagramPacket(receivebyte,receivebyte.length);
server.receive(receiver);
String str=new String(receiver.getData());
String s=str.trim();
//System.out.println(s);
InetAddress addr=receiver.getAddress();
int port=receiver.getPort();
String ip[]={"165.165.80.80","165.165.79.1"};
String name[]={"www.aptitudeguru.com","www.downloadcyclone.blogspot.com"};
for(int i=0;i<ip.length;i++)
{
if(s.equals(ip[i]))
{
sendbyte=name[i].getBytes();
DatagramPacket sender=new DatagramPacket(sendbyte,sendbyte.length,addr,port);
server.send(sender);
break;
}
else if(s.equals(name[i]))
{
sendbyte=ip[i].getBytes();
DatagramPacket sender=new DatagramPacket(sendbyte,sendbyte.length,addr,port);
server.send(sender);
break;
}
}
break;
}
}
catch(Exception e)
{
System.out.println(e);
}
}
}
```

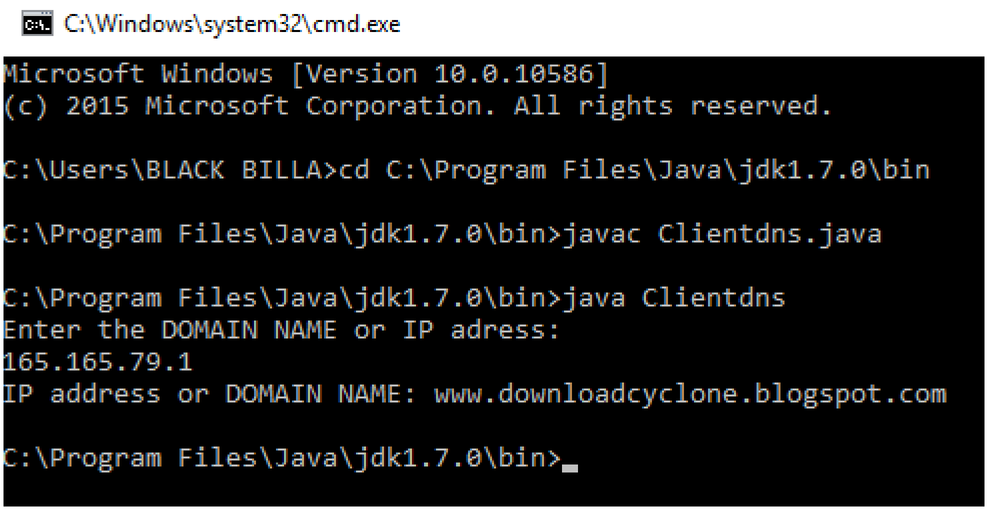
CLIENT PROGRAM

```
import java.io.*;
import java.net.*;
import java.util.*;
class Clientdns
{
public static void main(String args[])
{
try
{
DatagramSocket client=new DatagramSocket();
InetAddress addr=InetAddress.getByName("127.0.0.1");
byte[] sendbyte=new byte[1024];
byte[] receivebyte=new byte[1024];
BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter the DOMAIN NAME or IP adress:");
String str=in.readLine();
sendbyte=str.getBytes();
DatagramPacket sender=new DatagramPacket(sendbyte,sendbyte.length,addr,1309);
client.send(sender);
DatagramPacket receiver=new DatagramPacket(receivebyte,receivebyte.length);
client.receive(receiver);
String s=new String(receiver.getData());
System.out.println("IP address or DOMAIN NAME: "+s.trim());
client.close();
}
catch(Exception e)
{
System.out.println(e);
}}}
```

SERVER AND CLIENT OUTPUT:



```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac Serverdns.java

C:\Program Files\Java\jdk1.7.0\bin>java Serverdns

C:\Program Files\Java\jdk1.7.0\bin>
```



```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac Clientdns.java

C:\Program Files\Java\jdk1.7.0\bin>java Clientdns
Enter the DOMAIN NAME or IP adress:
165.165.79.1
IP address or DOMAIN NAME: www.downloadcyclone.blogspot.com

C:\Program Files\Java\jdk1.7.0\bin>
```

RESULT :

Thus the DNS application program was executed

Ex.No:5(a) **Write a code simulating ARP protocol**

DATE:

AIM:

To write a java code simulating ARP protocol

ALGORITHM:

Client

1. Start the program
2. Create socket and establish connection with the server.
3. Get the IP address to be converted into MAC address from the user.
4. Send this IP address to server.
5. Receive the MAC address for the IP address from the server.
6. Display the received MAC address
7. Terminate the connection

Server

1. Start the program
2. Create the socket, bind the socket created with IP address and port number and make it alistening socket.
3. Accept the connection request when it is requested by the client.
4. Server maintains the table in which IP and corresponding MAC addresses arestored.
5. Receive the IP address sent by the client.
6. Retrieve the corresponding MAC address for the IP address and send it to the client.
7. Close the connection with the client and now the server becomes a listening serverwaiting for the connection request from other clients
8. Stop

PROGRAM

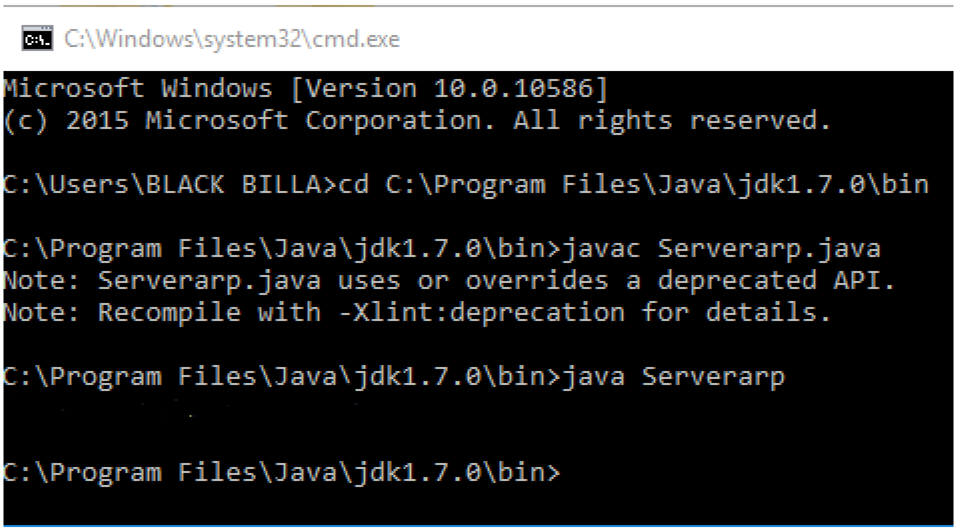
Server :

```
import java.io.*;
import java.net.*;
import java.util.*;
class Serverarp
{
 public static void main(String args[])
 {
 try
 {
 ServerSocket obj=new ServerSocket(139);
 Socket obj1=obj.accept();
 while(true)
 {
 DataInputStream din=new DataInputStream(obj1.getInputStream());
 DataOutputStream dout=new DataOutputStream(obj1.getOutputStream());
 String str=din.readLine();
 String ip[]={"165.165.80.80","165.165.79.1"};
 String mac[]={"6A:08:AA:C2","8A:BC:E3:FA"};
 for(int i=0;i<ip.length;i++)
 {
if(str.equals(ip[i]))
 {
 dout.writeBytes(mac[i]+'\n');
break;
 }
 }
 obj.close();
 }}
 catch(Exception e)
 {
 System.out.println(e);
 }}}
```

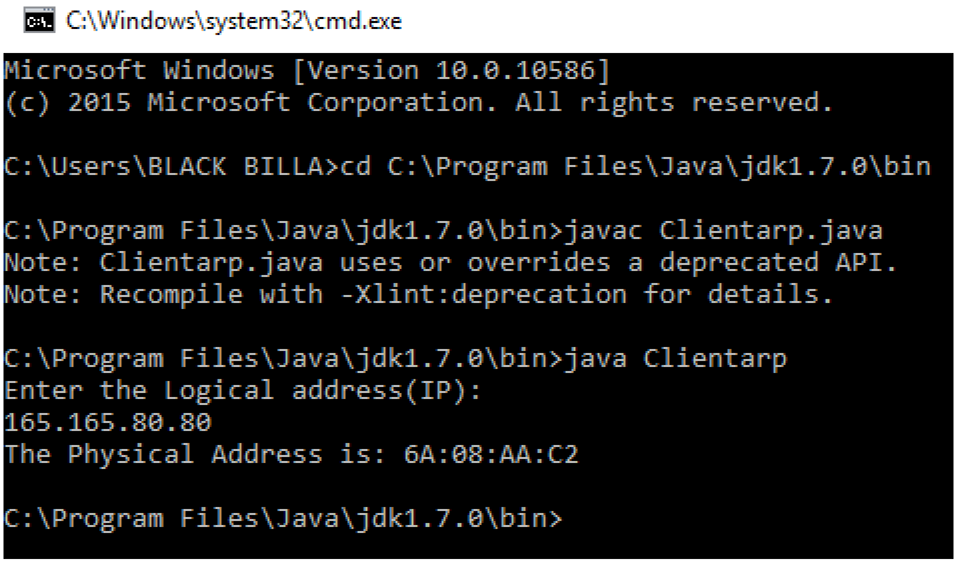
Client :

```
import java.io.*;
import java.net.*;
import java.util.*;
class Clientarp
{
 public static void main(String args[])
 {
 try
{
 BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
 Socket clsct=new Socket("127.0.0.1",139);
 DataInputStream din=new DataInputStream(clsct.getInputStream());
 DataOutputStream dout=new DataOutputStream(clsct.getOutputStream());
 System.out.println("Enter the Logical address(IP):");
 String str1=in.readLine();
 dout.writeBytes(str1+'\n');
 String str=din.readLine();
 System.out.println("The Physical Address is: "+str);
 clsct.close();
 }
 catch (Exception e)
 {
 System.out.println(e);
 }
 }
}
```

SERVER AND CLIENT OUTPUT :



```
C:\Windows\system32\cmd.exe

Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac Serverarp.java
Note: Serverarp.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Program Files\Java\jdk1.7.0\bin>java Serverarp


C:\Program Files\Java\jdk1.7.0\bin>
```



```
C:\Windows\system32\cmd.exe

Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac Clientarp.java
Note: Clientarp.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Program Files\Java\jdk1.7.0\bin>java Clientarp
Enter the Logical address(IP):
165.165.80.80
The Physical Address is: 6A:08:AA:C2

C:\Program Files\Java\jdk1.7.0\bin>
```

RESULT :

Thus the program has been executed successfully.

Ex.No:5(b) **simulating of RARP protocol**

DATE:

AIM:

To cricuit java code simulating RARP protocol

ALGORITHM:

Client

1. Start the program
2. Create datagram socket
3. Get the MAC address to be converted into IP address from the user.
4. Send this MAC address to server using UDP datagram.
5. Receive the datagram from the server and display the corresponding IP address.
6. Stop

Server

1. Start the program.
2. Server maintains the table in which IP and corresponding MAC addresses arestored.
3. Create the datagram socket
4. Receive the datagram sent by the client and read the MAC address sent.
5. Retrieve the IP address for the received MAC address from the table.
6. Display the corresponding IP address.
7. Stop

PROGRAM

Server :

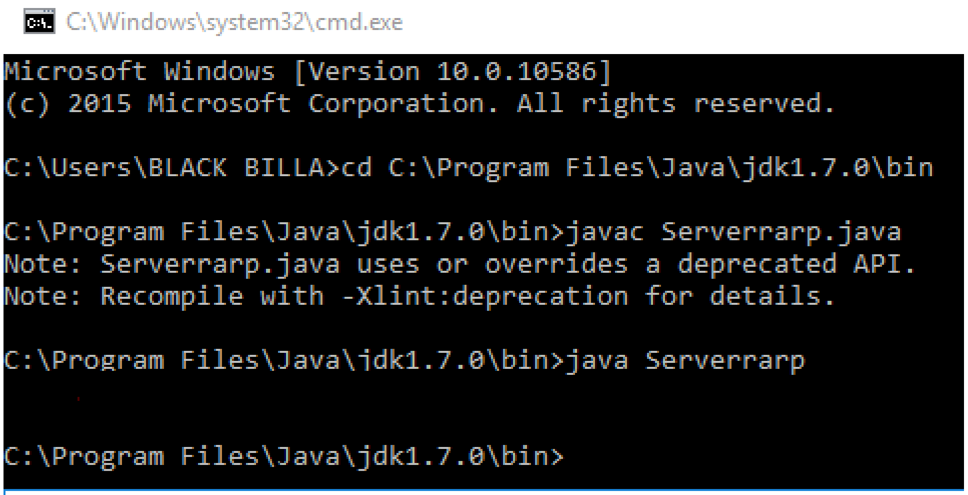
```
import java.io.*;
import java.net.*;
import java.util.*;
class Serverrarp
{
public static void main(String args[])
{
try
{
ServerSocket obj=new ServerSocket(139);
Socket obj1=obj.accept();
while(true)
{
DataInputStream din=new DataInputStream(obj1.getInputStream());
DataOutputStream dout=new DataOutputStream(obj1.getOutputStream());
String str=din.readLine();
String ip[]={"165.165.80.80","165.165.79.1"};
String mac[]={"6A:08:AA:C2","8A:BC:E3:FA"};
for(int i=0;i<mac.length;i++)
{
if(str.equals(mac[i]))
{
dout.writeBytes(ip[i]+'\n');
break;
}}
obj.close();
}}
catch(Exception e)
{
System.out.println(e);
}
```

```
        }}
```

Client :

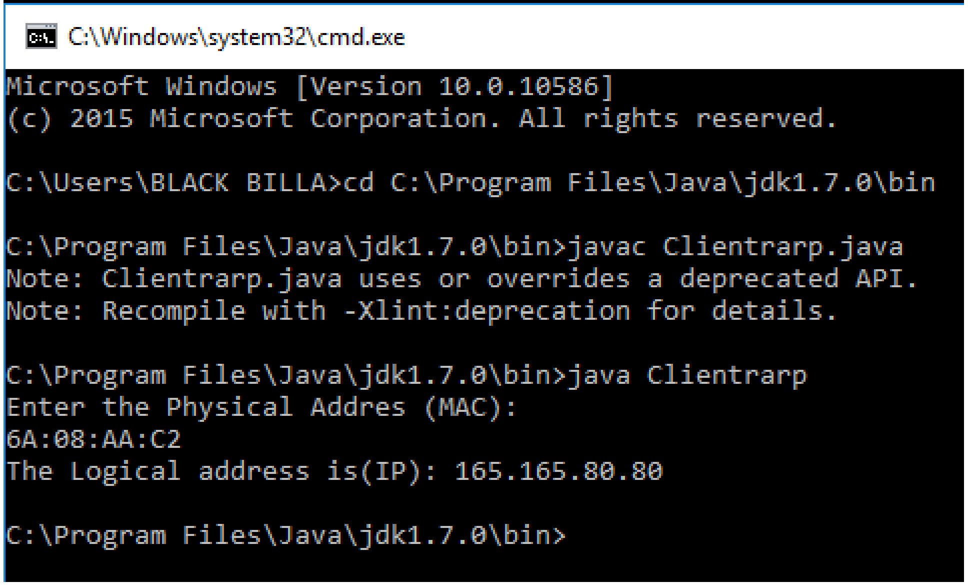
```
import java.io.*;
import java.net.*;
import java.util.*;
class Clientrarp
{
public static void main(String args[])
{
try
{
BufferedReader in=new BufferedReader(new InputStreamReader(System.in));
Socket clsct=new Socket("127.0.0.1",139);

DataInputStream din=new DataInputStream(clsct.getInputStream());
DataOutputStream dout=new DataOutputStream(clsct.getOutputStream());
System.out.println("Enter the Physical Addres (MAC):");
String str1=in.readLine();
dout.writeBytes(str1+'\n');
String str=din.readLine();
System.out.println("The Logical address is(IP): "+str);
clsct.close();
}
catch (Exception e)
{
System.out.println(e);
}
}
}
```

SERVER AND CLIENT OUTPUT :



C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac Serverrarp.java
Note: Serverrarp.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Program Files\Java\jdk1.7.0\bin>java Serverrarp


C:\Program Files\Java\jdk1.7.0\bin>
```



C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac Clientrarp.java
Note: Clientrarp.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.

C:\Program Files\Java\jdk1.7.0\bin>java Clientrarp
Enter the Physical Addres (MAC):
6A:08:AA:C2
The Logical address is(IP): 165.165.80.80

C:\Program Files\Java\jdk1.7.0\bin>
```

RESULT:

Thus the program has been executed successfully

EX NO 6

STUDY OF NETWORK SIMULATOR (NS) AND SIMULATION OF CONGESTION CONTROL ALGORITHMS USING NS

AIM:

To Study of Network simulator (NS).and Simulation of Congestion Control Algorithms using NS.

NET WORK SIMULATOR (NS2)

Ns overview

- ✓ Ns programming: A Quick start
- ✓ Case study I: A simple Wireless network
- ✓ Case study II: Create a new agent in Ns

Ns overview

- ✓ Ns Status
- ✓ Periodical release (ns-2.26, Feb 2003)
- ✓ Platform support
- ✓ FreeBSD, Linux, Solaris, Windows and Mac

Ns Functionalities

Routing, Transportation, Traffic sources, Queuing disciplines, QoS

Wireless

Ad hoc routing, mobile IP, sensor-MAC
Tracing, visualization and various utilities
NS(Network Simulators)

Most of the commercial simulators are GUI driven, while some network simulators are CLI driven. The network model / configuration describes the state of the network (nodes, routers, switches, links) and the events (data transmissions, packet error etc.). An important output of simulations are the trace files. Trace files log every packet, every event that occurred in the simulation and are used for analysis. Network simulators can also provide other tools to facilitate visual analysis of trends and potential trouble spots.

Most network simulators use discrete event simulation, in which a list of pending "events" is stored, and those events are processed in order, with some events triggering future events—such as the event of the arrival of a packet at one node triggering the event of the arrival of that packet at a downstream node.

Simulation of networks is a very complex task. For example, if congestion is high, then estimation of the average occupancy is challenging because of high variance. To estimate the likelihood of a buffer overflow in a network, the time required for an accurate answer can be extremely large. Specialized techniques such as "control variates" and "importance sampling" have been developed to speed simulation.

Examples of network simulators

There are many both free/open-source and proprietary network simulators. Examples of notable network simulation software are, ordered after how often they are mentioned in research papers:

1. ns (open source)
2. OPNET (proprietary software)
3. NetSim (proprietary software)

Uses of network simulators

Network simulators serve a variety of needs. Compared to the cost and time involved in setting up an entire test bed containing multiple networked computers, routers and data links, network simulators are relatively fast and inexpensive. They allow engineers, researchers to test scenarios that might be particularly difficult or expensive to emulate using real hardware - for instance, simulating a scenario with several nodes or experimenting with a new protocol in the network. Network simulators are particularly useful in allowing researchers to test new networking protocols or changes to existing protocols in a controlled and reproducible environment. A typical network simulator encompasses a wide range of networking technologies and can help the users to build complex networks from basic building blocks such as a variety of nodes and links. With the help of simulators, one can design hierarchical networks using various types of nodes like computers, hubs, bridges, routers, switches, links, mobile units etc.

Various types of Wide Area Network (WAN) technologies like TCP, ATM, IP etc. and Local Area Network (LAN) technologies like Ethernet, token rings etc., can all be simulated with a typical simulator and the user can test, analyze various standard results apart from devising some novel protocol or strategy for routing etc. Network simulators are also widely used to simulate battlefield networks in Network-centric warfare.

There are a wide variety of network simulators, ranging from the very simple to the very complex. Minimally, a network simulator must enable a user to represent a network topology, specifying the nodes on the network, the links between those nodes and the traffic between the nodes. More complicated systems may allow the user to specify everything about the protocols used to handle traffic in a network. Graphical applications allow users to easily visualize the workings of their simulated environment. Text-based applications may provide a less intuitive interface, but may permit more advanced forms of customization.

Packet loss

Packet loss occurs when one or more packets of data travelling across a computer network fail to reach their destination. Packet loss is distinguished as one of the three main error types encountered in digital communications; the other two being bit error and spurious packets caused due to noise.

Packets can be lost in a network because they may be dropped when a queue in the network node overflows. The amount of packet loss during the steady state is another important property of a congestion control scheme. The larger the value of packet loss, the more difficult it is for transport layer protocols to maintain high bandwidths, the sensitivity to loss of individual packets, as well as to frequency and patterns of loss among longer packet sequences is strongly dependent on the application itself.

Throughput

This is the main performance measure characteristic, and most widely used. In communication networks, such as Ethernet or packet radio, throughput or network throughput is the average rate of successful message delivery over a communication channel. The throughput is usually measured in bits per second (bit/s or bps), and sometimes in data packets per second or data packets per time slot. This measure how soon the receiver is able to get a certain amount of data send by the sender. It is determined as the ratio of the total data received to the end to end delay. Throughput is an important factor which directly impacts the network performance

Delay

Delay is the time elapsed while a packet travels from one point e.g., source premise or network ingress to destination premise or network degrees. The larger the value of delay, the more difficult it is for transport layer protocols to maintain high bandwidths. We will calculate end to end delay

Queue Length

A queuing system in networks can be described as packets arriving for service, waiting for service if it is not immediate, and if having waited for service, leaving the system after being served. Thus queue length is very important characteristic to determine that how well the active queue management of the congestion control algorithm has been working.

RESULT

The Study of Network simulator (NS) and Simulation of Congestion Control Algorithms using NS has been done successfully.

EX NO 7

STUDY OF TCP AND UDP PERFORMANCE

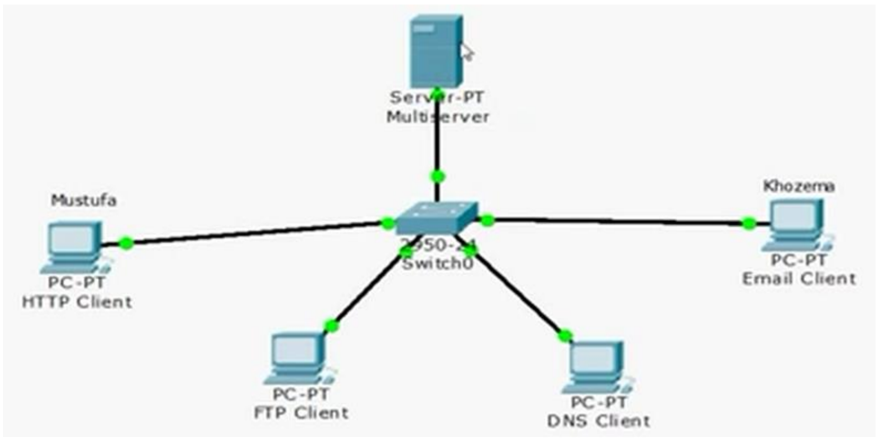
AIM

To make the study of tcp/udp performance using simulation tool.

PROCEDURE

DEPLOYING AND CONFIGURING THE TOPOLOGY

Step 1: Make the topology multiserver, dns client, email client, ftp client and http client as below as tools available in packet tracer.



Step 2: Double click the multiserver – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.0.2** – subnet 255.255.255.0 – default gateway – 192.168.0.1 – dns server 192.168.0.2.

Config – Choose http – make sure enable radio button on http and https – erase the html code and write the customized html code.

Config – Choose dns – make sure enable "on" radio button on dns service – name : sbc.edu.in – address : 192.168.0.2 – click add button.

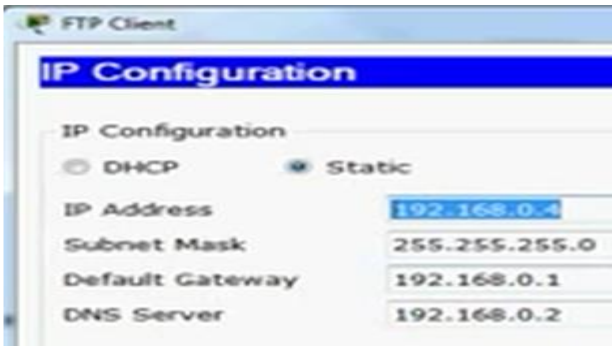
Config – Choose email – make sure enable "on" radio button on smtp and pop3 services – domain name : sbc.edu.in – add two different user name with password by clicking "+ button".

Config – Choose ftp – make sure enable "on" radio button on ftp services – provide user name and password for file access on user setup part – check all privileges – click "+ button" to add current ftp user.

Step 3 : Double click the http client – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.0.3** – subnet 255.255.255.0 – default gateway – 192.168.0.1 – dns server 192.168.0.2.

Desktop – choose e mail – your name : user name given in email server configuration – email address @ domain name for example : username1@sbc.edu.in – incoming and outgoing mail server as sbc.edu.in – provide user name and password that given in email server configuration and click “save” button – inbox will appears.

Step 4 : Double click the ftp client – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.0.4** – subnet 255.255.255.0 – default gateway – 192.168.0.1 – dns server 192.168.0.2.



FTP Client

IP Configuration

IP Configuration

DHCP [✓ Static]

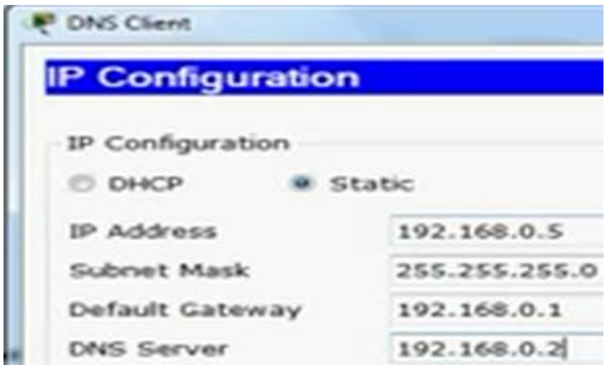
IP Address 192.168.0.4

Subnet Mask 255.255.255.0

Default Gateway 192.168.0.1

DNS Server 192.168.0.2

Step 5 : Double click the dns client – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.0.5** – subnet 255.255.255.0 – default gateway – 192.168.0.1 – dns server 192.168.0.2.



DNS Client

IP Configuration

IP Configuration

DHCP [✓ Static]

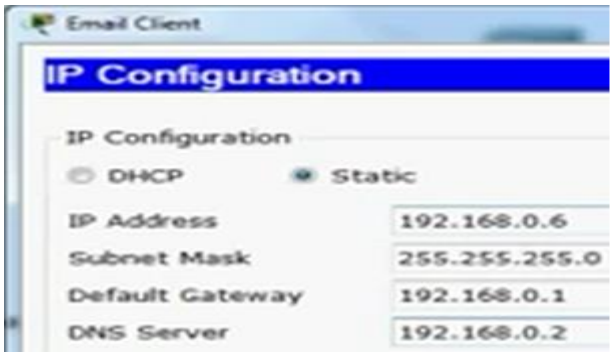
IP Address 192.168.0.5

Subnet Mask 255.255.255.0

Default Gateway 192.168.0.1

DNS Server 192.168.0.2

Step 6 : Double click the email client – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.0.6** – subnet 255.255.255.0 – default gateway – 192.168.0.1 – dns server 192.168.0.2.



Email Client

IP Configuration

IP Configuration

DHCP [✓ Static]

IP Address 192.168.0.6

Subnet Mask 255.255.255.0

Default Gateway 192.168.0.1

DNS Server 192.168.0.2

Desktop – choose e mail – your name : username2 given in email server configuration – email address @ domain name for example : username2@sbc.edu.in – incoming and outgoing mail server as sbc.edu.in – provide user name and password that given in email server configuration and click “save” button – inbox will appears.

GENERATING NETWORK TRAFFICS

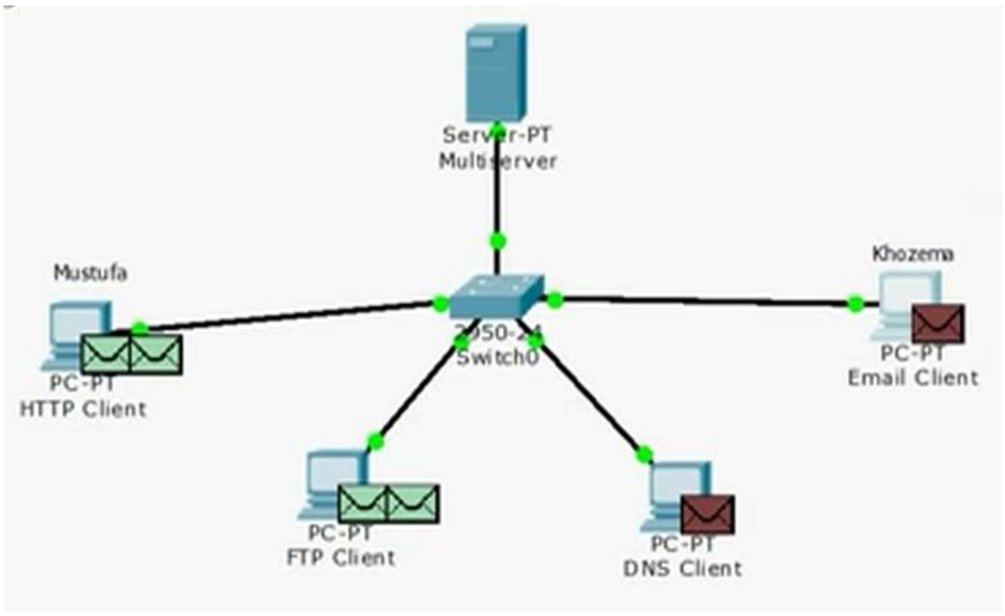
Step 6 : Generate traffic to populate address resolution protocol (arp) tables, click multiserver – click desktop – commamd prompt – enter the ping 192.168.0.255 command.

Step 7 : Generate web (http) traffic – switch to simulation mode – click http client – click desktop – web browser – enter 192.168.0.2 in url field – click go – pdu will appear – don’t close the current window just minimize it.

Step 8 : Generate ftp traffic – stay with simulation mode – click ftp client – click desktop – commamd prompt – enter the ftp 192.168.0.2 command – pdu will appear – don’t close the current window just minimize it.

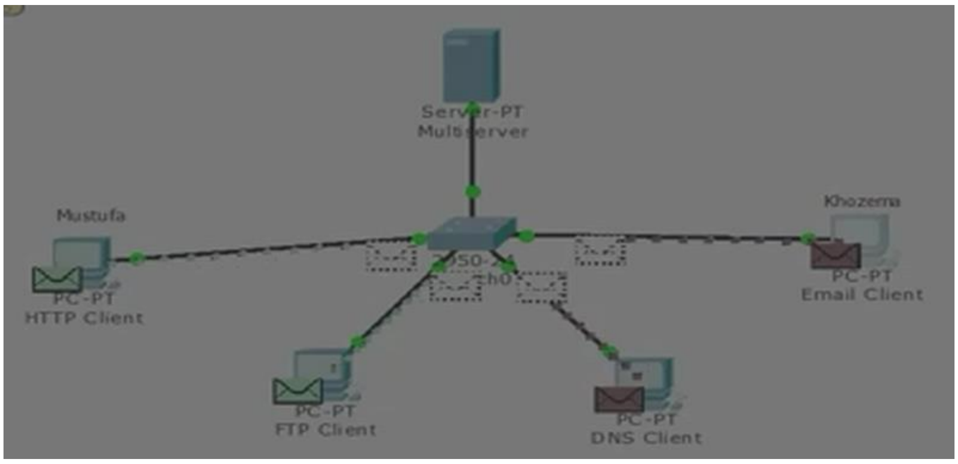
Step 9 : Generate dns traffic – stay with simulation mode – click dns client – click desktop – commamd prompt – enter the nslookup sbc.edu.in command – pdu will appear – don’t close the current window just minimize it.

Step 10 : Generate email traffic – stay with simulation mode – click email client – click desktop – email – inbox will appear – compose a mail to user1 with user1’s mail id – click send button – pdu will appear – don’t close the current window just minimize it.

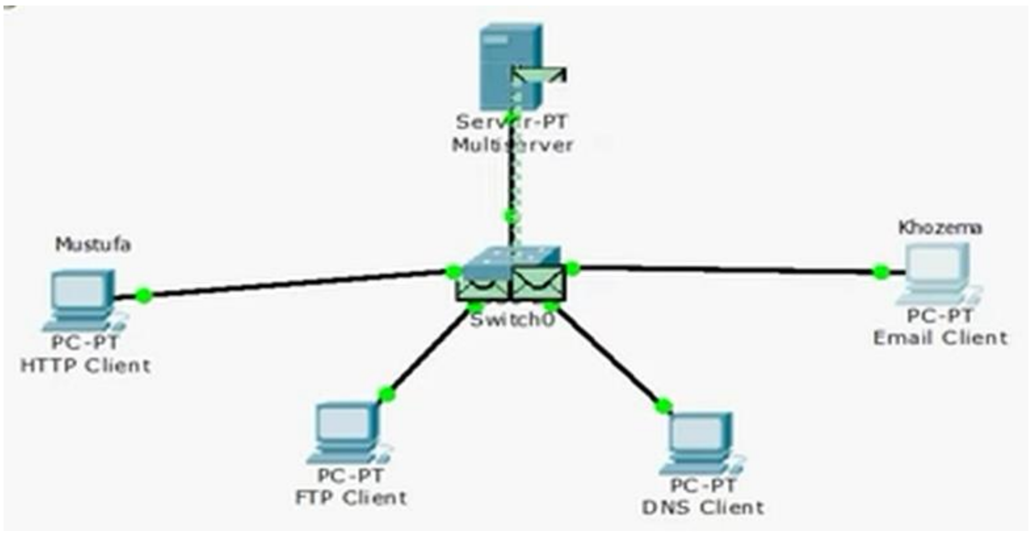


EXAMINING THE FUNCTIONALITY OF TCP AND UDP

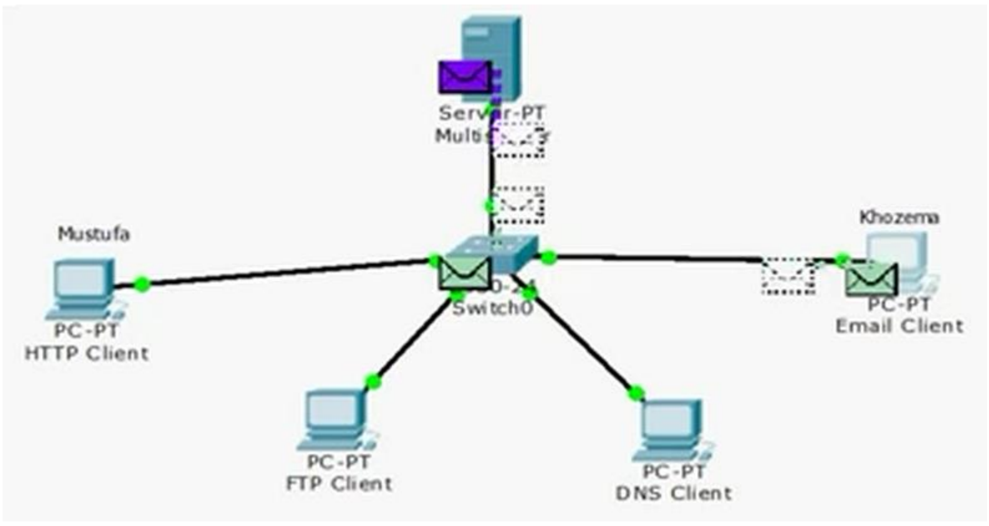
Step 11 : Now corresponding traffics are generated, all client computers has pdu list and ready for simulation. Click capture / forward only once, all of the pdus are transferred to switch.



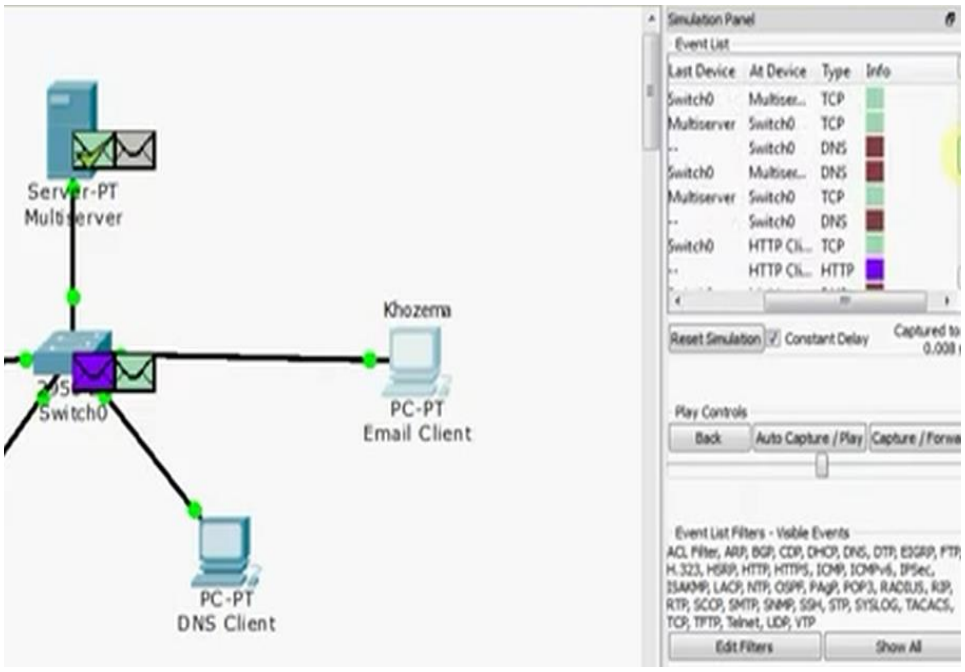
Step 12 : Click capture / forward again, some of the pdus are disappear as they are saved in th queue.



Step 13 : Click capture / forward six times, all the clients received the reply.



Step 14 : Finally by the way the study and examining of tcp / udp is visualised in packet tracer as shown in the following diagram.



Last Device	At Device	Type	Info
Switch0	Multiser...	TCP	🟩
Multiserver	Switch0	TCP	🟩
--	Switch0	DNS	🟥
Switch0	Multiser...	DNS	🟥
Multiserver	Switch0	TCP	🟩
--	Switch0	DNS	🟥
Switch0	HTTP Cli...	TCP	🟩
--	HTTP Cli...	HTTP	🟪

ACL Filter, ARP, BGP, CDP, DHCP, DNS, DTP, EIGRP, FTP, H.323, HSRP, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NTP, OSPF, PAgP, POP3, RADIUS, RIP, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

RESULT :

The study of tcp/udp performance using simulation tool has been done successfully.

Ex.No: 8(a) Simulation of Distance Vector Routing algorithm.

DATE :

AIM:

TO write a java program for simulating the distance vector algorithm

ALGORITHM:

1. Create a Simulator object.
2. Set routing as dynamic.
3. Open the trace and nam trace files.
4. Define the finish procedure.
5. Create nodes and the links between them.
6. Create the agents and attach them to the nodes.
7. Create the applications and attach them to the udp agent.
8. Connect udp and null..
9. At 1 sec the link between node 1 and 2 is broken.
10. At 2 sec the link is up again.
11. Run the simulation.

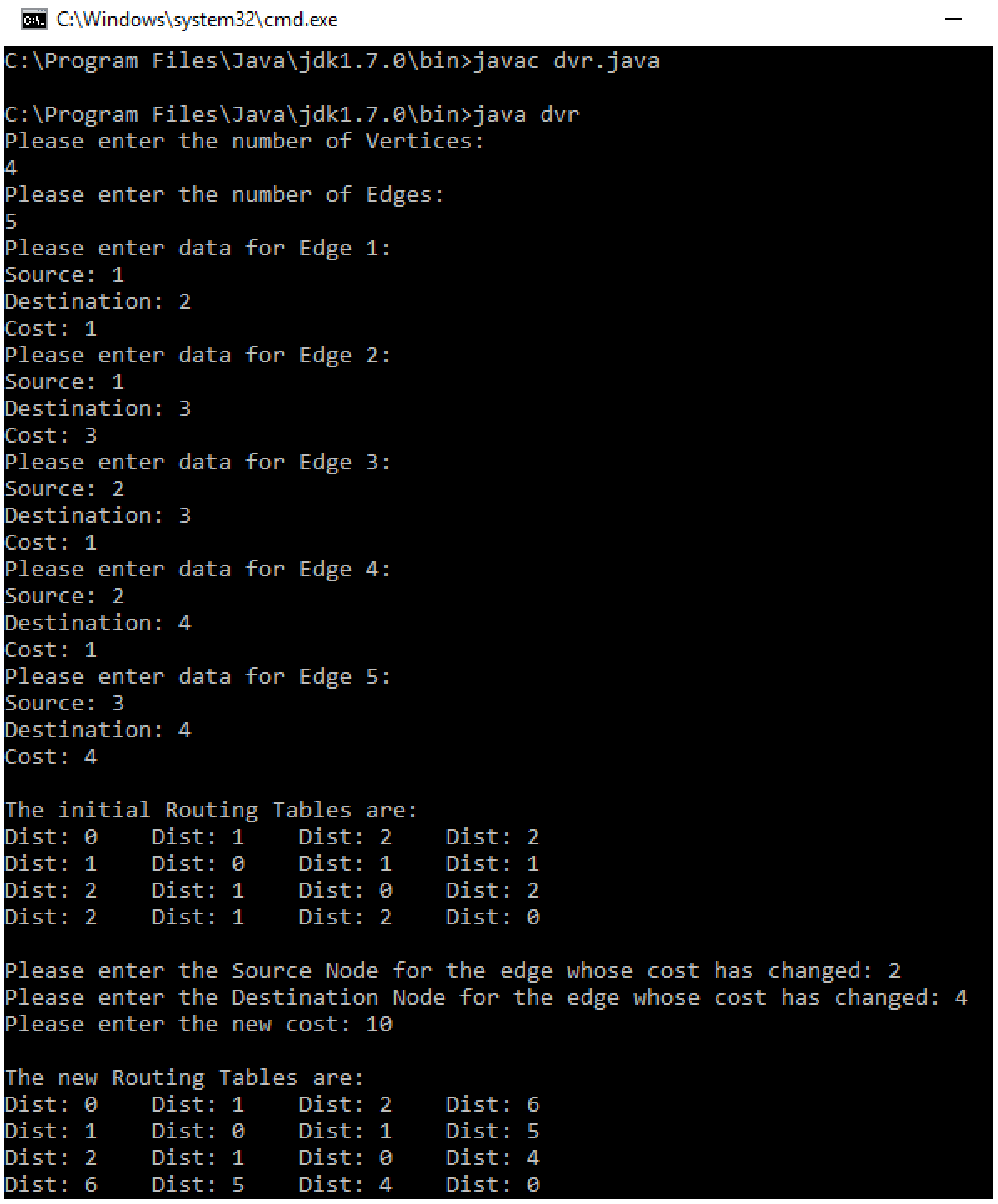
PROGRAM :

```
import java.io.*;
public class dvr
{
 static int graph[][];
 static int via[][];
 static int rt[][];
 static int v;
 static int e;
 public static void main(String args[]) throws IOException
 {
  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
  System.out.println("Please enter the number of Vertices: ");
  v = Integer.parseInt(br.readLine());
  System.out.println("Please enter the number of Edges: ");
  e = Integer.parseInt(br.readLine());
  graph = new int[v][v];
  via = new int[v][v];
  rt = new int[v][v];
  for(int i = 0; i < v; i++)
   for(int j = 0; j < v; j++)
   {
    if(i == j)
     graph[i][j] = 0;
    else
     graph[i][j] = 9999;
   }
  for(int i = 0; i < e; i++)
  {
   System.out.println("Please enter data for Edge " + (i + 1) + ":");
   System.out.print("Source: ");
   int s = Integer.parseInt(br.readLine());
   s--;
   System.out.print("Destination: ");
   int d = Integer.parseInt(br.readLine());
   d--;
   System.out.print("Cost: ");
   int c = Integer.parseInt(br.readLine());
   graph[s][d] = c;
   graph[d][s] = c;
  }
  dvr_calc_disp("The initial Routing Tables are: ");
  System.out.print("Please enter the Source Node for the edge whose cost has
changed: ");
  int s = Integer.parseInt(br.readLine());
  s--;
  System.out.print("Please enter the Destination Node for the edge whose cost has
changed: ");
  int d = Integer.parseInt(br.readLine());
  d--;
  System.out.print("Please enter the new cost: ");
  int c = Integer.parseInt(br.readLine());
  graph[s][d] = c;
  graph[d][s] = c;
  dvr_calc_disp("The new Routing Tables are: ");
```

```
}
static void dvr_calc_disp(String message)
{
 System.out.println();
 init_tables();
 update_tables();
 System.out.println(message);
 print_tables();
 System.out.println();
}
static void update_table(int source)
{
 for(int i = 0; i < v; i++)
 {
  if(graph[source][i] != 9999)
  {
   int dist = graph[source][i];
   for(int j = 0; j < v; j++)
   {
    int inter_dist = rt[i][j];
    if(via[i][j] == source)
     inter_dist = 9999;
    if(dist + inter_dist < rt[source][j])
    {
     rt[source][j] = dist + inter_dist;
     via[source][j] = i;
    }
   }
  }
 }
}
static void update_tables()
{
 int k = 0;
 for(int i = 0; i < 4*v; i++)
 {
  update_table(k);
  k++;
  if(k == v)
   k = 0;
 }
}
static void init_tables()
{
 for(int i = 0; i < v; i++)
 {
  for(int j = 0; j < v; j++)
  {
   if(i == j)
   {
    rt[i][j] = 0;
    via[i][j] = i;
   }
   else
```

```
    {
     rt[i][j] = 9999;
     via[i][j] = 100;
    }
   }
  }
 }
 static void print_tables()
 {
  for(int i = 0; i < v; i++)
  {
   for(int j = 0; j < v; j++)
   {
    System.out.print("Dist: " + rt[i][j] + "    ");
   }
   System.out.println();
  }
 }
}
```

OUTPUT :


C:\Windows\system32\cmd.exe

```
C:\Program Files\Java\jdk1.7.0\bin>javac dvr.java

C:\Program Files\Java\jdk1.7.0\bin>java dvr
Please enter the number of Vertices:
4
Please enter the number of Edges:
5
Please enter data for Edge 1:
Source: 1
Destination: 2
Cost: 1
Please enter data for Edge 2:
Source: 1
Destination: 3
Cost: 3
Please enter data for Edge 3:
Source: 2
Destination: 3
Cost: 1
Please enter data for Edge 4:
Source: 2
Destination: 4
Cost: 1
Please enter data for Edge 5:
Source: 3
Destination: 4
Cost: 4

The initial Routing Tables are:
Dist: 0    Dist: 1    Dist: 2    Dist: 2
Dist: 1    Dist: 0    Dist: 1    Dist: 1
Dist: 2    Dist: 1    Dist: 0    Dist: 2
Dist: 2    Dist: 1    Dist: 2    Dist: 0

Please enter the Source Node for the edge whose cost has changed: 2
Please enter the Destination Node for the edge whose cost has changed: 4
Please enter the new cost: 10

The new Routing Tables are:
Dist: 0    Dist: 1    Dist: 2    Dist: 6
Dist: 1    Dist: 0    Dist: 1    Dist: 5
Dist: 2    Dist: 1    Dist: 0    Dist: 4
Dist: 6    Dist: 5    Dist: 4    Dist: 0
```

RESULT:

Thus the program has been executed successf

Ex.No: 8(b) Simulation of Distance link state Routing algorithm.

DATE :

AIM:

To write a java code for simulating the link state routing algorithm

ALGORITHM:

1. Create a simulator object
2. Set routing protocol to Distance Vector routing
3. Trace packets on all links onto NAM trace and text trace file
4. Define finish procedure to close files, flush tracing and run NAM
5. Create eight nodes
6. Specify the link characteristics between nodes
7. Describe their layout topology as a octagon
8. Add UDP agent for node n1
9. Create CBR traffic on top of UDP and set traffic parameters.
10. Add a sink agent to node n4
11. Connect source and the sink
12. Schedule events as follows:
    a. Start traffic flow at 0.5
    b. Down the link n3-n4 at 1.0
    c. Up the link n3-n4 at 2.0
    d. Stop traffic at 3.0
    e. Call finish procedure at 5.0
13. Start the scheduler
14. Observe the traffic route when link is up and down
15. View the simulated events and trace file analyze it
16. Stop

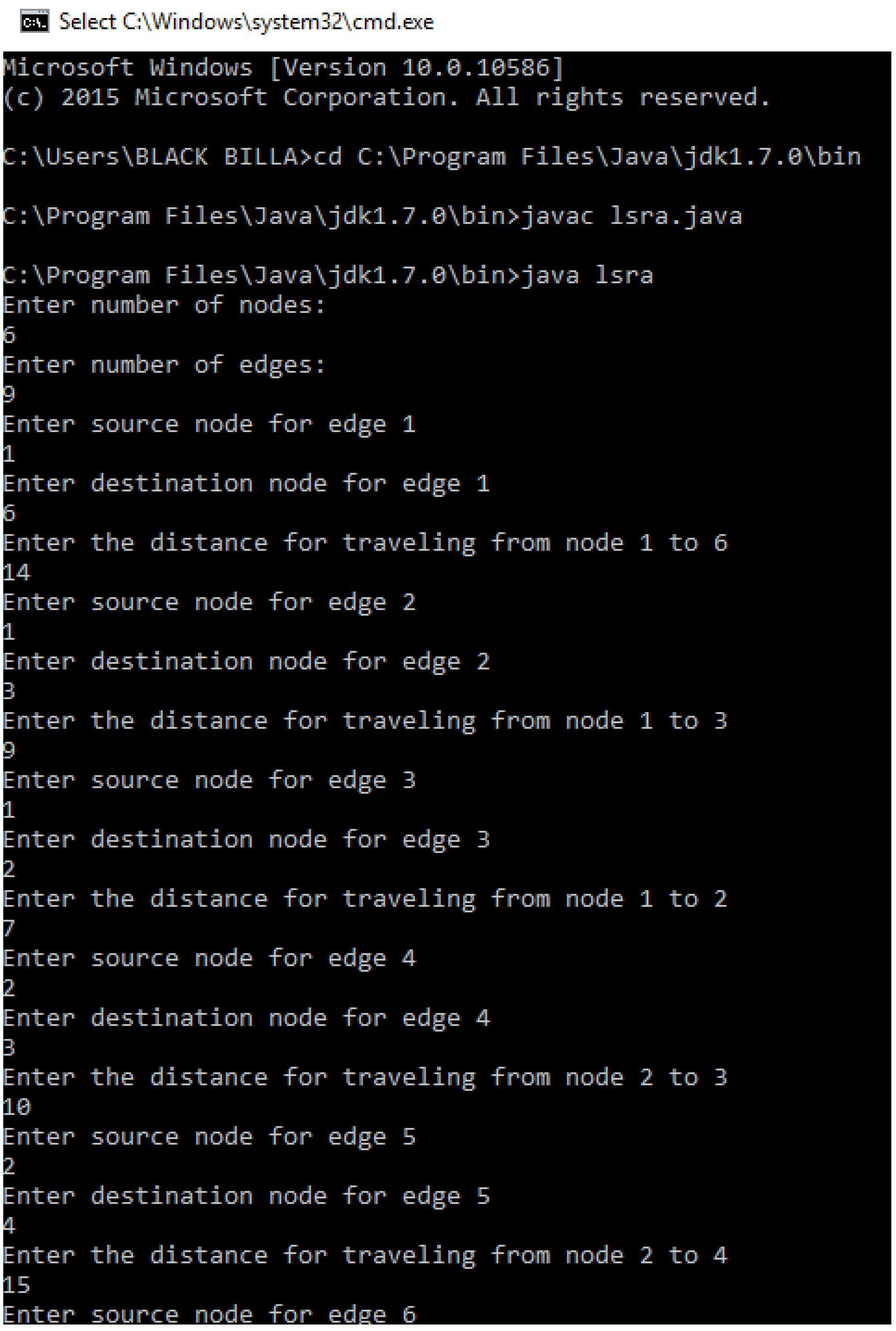
PROGRAM :

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
class Graph
{
 int nodes;
 int edges;
 int grph[][];
 void accept() throws IOException
 {
  BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
  System.out.println("Enter number of nodes: ");
  nodes = Integer.parseInt(br.readLine());
  System.out.println("Enter number of edges: ");
  edges = Integer.parseInt(br.readLine());
  grph = new int[nodes][nodes];
  for(int i = 0; i < nodes; i++)
   for(int j = 0; j < nodes; j++)
    grph[i][j] = 9999;
  for(int i = 0; i < edges; i++)
  {
   System.out.println("Enter source node for edge " + (i + 1));
   int source = Integer.parseInt(br.readLine());
   System.out.println("Enter destination node for edge " + (i + 1));
   int destination = Integer.parseInt(br.readLine());
   System.out.println("Enter the distance for traveling from node " + source + " to " +
destination);
   int distance = Integer.parseInt(br.readLine());
   grph[source - 1][destination - 1] = distance;
   grph[destination - 1][source - 1] = distance;
  }
 }
}
class lsra {
 static void dijk(Graph g, int src)
 {
  int distance[] = new int[g.nodes];
  int previous[] = new int[g.nodes];
  String path[] = new String[g.nodes];
  for(int i = 0; i < g.nodes; i++)
  {
   distance[i] = 9999;
   previous[i] = -1;
   path[i] = "";
  }
  distance[src] = 0;
  for(int i = 0; i < g.nodes; i++)
  {
   for(int j = 0; j < g.nodes; j++)
   {
    if(distance[i] + g.grph[i][j] < distance[j])
    {
```

```
    distance[j] = distance[i] + g.grph[i][j];
    previous[j] = i;
   }
  }
 }
 for(int i = 0; i < g.nodes; i++)
 {
  int current = i;
  do
  {
   path[i] = "-" + (current + 1) + path[i];
   current = previous[current];
  }
  while(current != -1);
  path[i] = path[i].substring(1, path[i].length());
 }
 System.out.println();
 for(int i = 0; i < g.nodes; i++)
 {
  if(i != src)
  {
   System.out.println("The shortest path & its distance for node " + (i + 1) + " is:");
   System.out.print("Path: " + path[i]);
   System.out.println(" Distance: " + distance[i]);
  }
 }
}
```

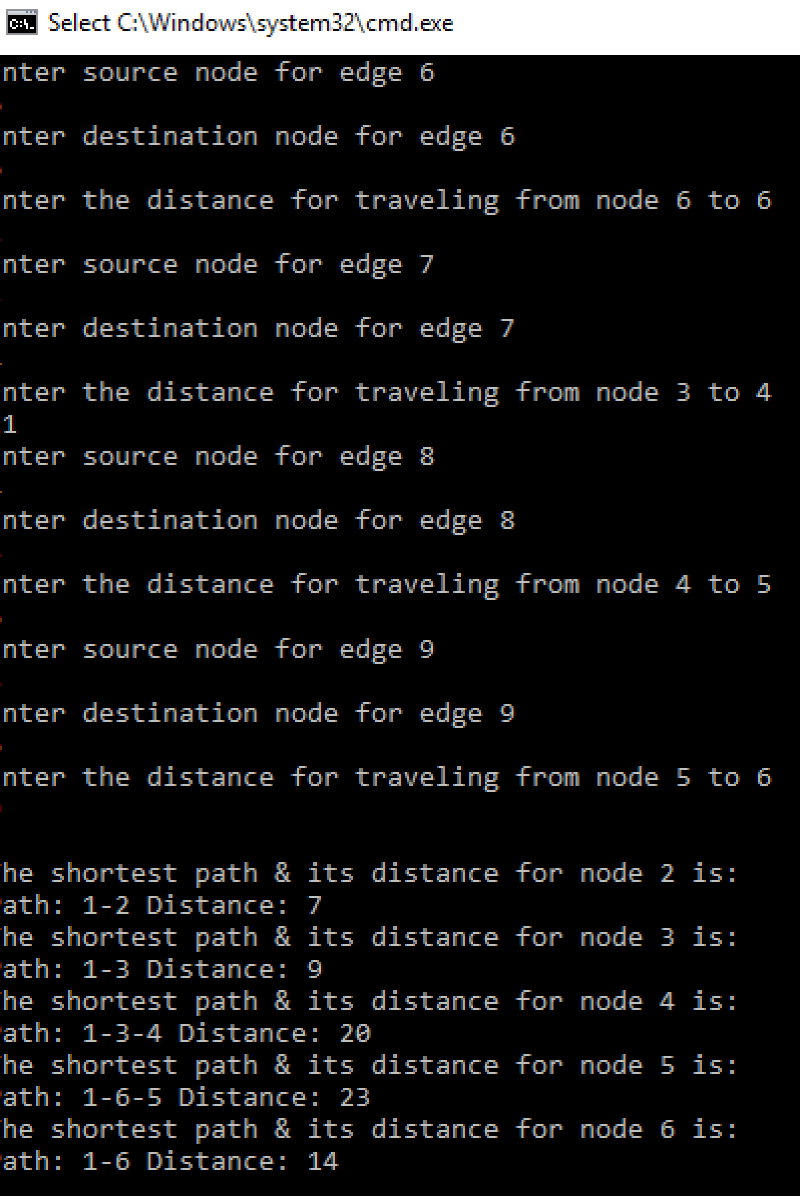
```
 public static void main(String args[]) throws IOException
 {
  Graph g = new Graph();
  g.accept();
  dijk(g, 0);
 }
}
```

OUTPUT :



Select C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\BLACK BILLA>cd C:\Program Files\Java\jdk1.7.0\bin

C:\Program Files\Java\jdk1.7.0\bin>javac lsra.java

C:\Program Files\Java\jdk1.7.0\bin>java lsra
Enter number of nodes:
6
Enter number of edges:
9
Enter source node for edge 1
1
Enter destination node for edge 1
6
Enter the distance for traveling from node 1 to 6
14
Enter source node for edge 2
1
Enter destination node for edge 2
3
Enter the distance for traveling from node 1 to 3
9
Enter source node for edge 3
1
Enter destination node for edge 3
2
Enter the distance for traveling from node 1 to 2
7
Enter source node for edge 4
2
Enter destination node for edge 4
3
Enter the distance for traveling from node 2 to 3
10
Enter source node for edge 5
2
Enter destination node for edge 5
4
Enter the distance for traveling from node 2 to 4
15
Enter source node for edge 6
```


Select C:\Windows\system32\cmd.exe

```
Enter source node for edge 6
6
Enter destination node for edge 6
6
Enter the distance for traveling from node 6 to 6
2
Enter source node for edge 7
3
Enter destination node for edge 7
4
Enter the distance for traveling from node 3 to 4
11
Enter source node for edge 8
4
Enter destination node for edge 8
5
Enter the distance for traveling from node 4 to 5
6
Enter source node for edge 9
5
Enter destination node for edge 9
6
Enter the distance for traveling from node 5 to 6
9

The shortest path & its distance for node 2 is:
Path: 1-2 Distance: 7
The shortest path & its distance for node 3 is:
Path: 1-3 Distance: 9
The shortest path & its distance for node 4 is:
Path: 1-3-4 Distance: 20
The shortest path & its distance for node 5 is:
Path: 1-6-5 Distance: 23
The shortest path & its distance for node 6 is:
Path: 1-6 Distance: 14
```

RESULT :

The simulation of link state routing algorithm using java programming has been done successfully

EX NO 9

PERFORMANCE EVALUVATION OF ROUTING PROTOCOLS

AIM

To make the analysis of performance evaluation of routing protocols using simulation tool.

PROCEDURE

RIP – ROUTING INFORMATION PROTOCOL

RIP is a commonly used routing protocol in small to medium TCP/IP networks. Routing Information Protocol (RIP) is a stable protocol that uses a distance-vector algorithm to calculate routes.

Routing Information Protocol (RIP) uses hop count as the metric to rate the value of different routes. The hop count is the number of devices that can be traversed in a route. A directly connected network has a metric of zero; an unreachable network has a metric of 16. This limited metric range makes RIP unsuitable for large networks. If a device does not receive an update from another device for 180 seconds or more, the receiving device marks the routes served by the non-updating device as unusable. If there is still no update after 240 seconds, the device removes all routing table entries for the non-updating device.

A device that is running RIP can receive a default network via an update from another device that is running RIP, or the device can source the default network using RIP. In both cases, the default network is advertised through RIP to other RIP neighbours.

The Routing Information Protocol (RIP) sends routing-update messages at regular intervals and when the network topology changes. When a device receives a RIP routing update that includes changes to an entry, the device updates its routing table to reflect the new route. The metric value for the path is increased by 1, and the sender is indicated as the next hop. RIP devices maintain only the best route (the route with the lowest metric value) to a destination. After updating its routing table, the device immediately begins transmitting RIP routing updates to inform other network devices of the change. These updates are sent independently of the regularly scheduled updates that RIP devices send.

The Routing Information Protocol (RIP) uses a single routing metric to measure the distance between the source and the destination network. Each hop in a path from the source to the destination is assigned a hop-count value, which is typically 1. When a device receives a routing update that contains a new or changed destination network entry, the device adds 1 to the metric value indicated in the update and enters the network in the routing table. The IP address of the sender is used as the next hop. If an interface network is not specified in the routing table, it will not be advertised in any RIP update.

Routing Information Protocol (RIP) is normally a broadcast protocol, and for RIP routing updates to reach non-broadcast networks, there must be configure the router software to permit this exchange of routing information.

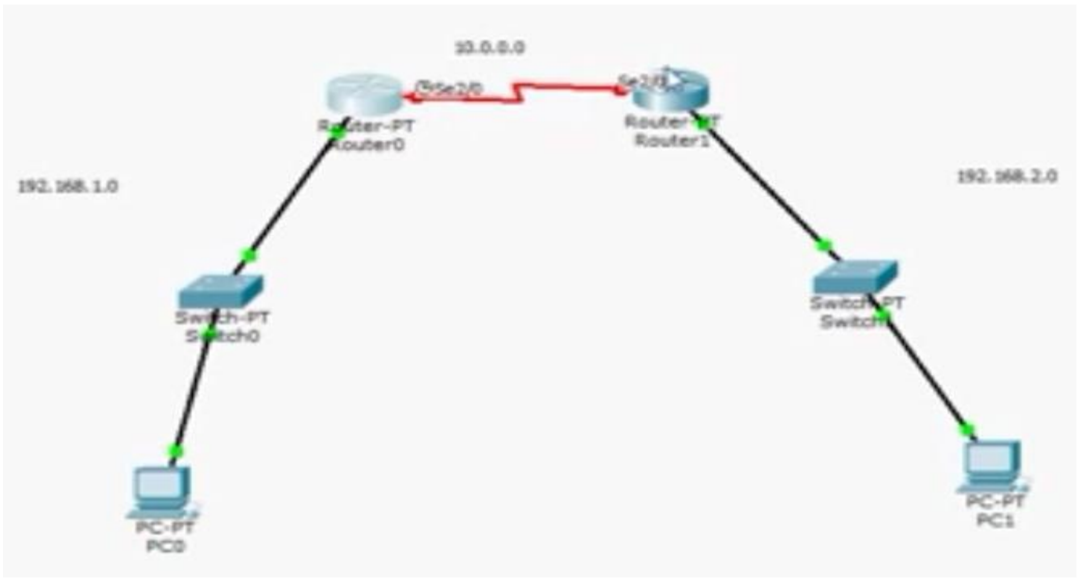
To control the set of interfaces to exchange routing updates, enable or disable the sending of routing updates on specified interfaces by configuring the passive-interface router configuration command.

There is possibility to use an offset list to increase increasing incoming and outgoing metrics to routes learned via RIP. Optionally, limit the offset list with either an access list or an interface. Routing protocols use several timers that determine variables such as the frequency of routing updates, the length of time before a route becomes invalid, and other parameters. Network administrator can adjust these timers to tune routing protocol performance to better suit your internetwork needs. The possible timer adjustments:

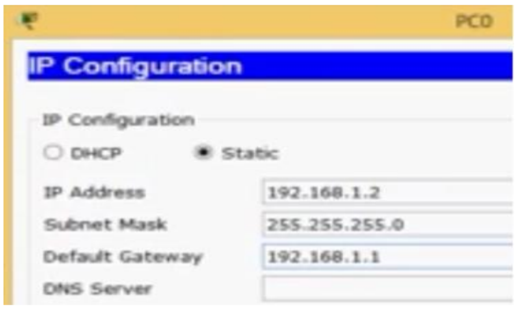
- ✓ The rate (time, in seconds, between updates) at which routing updates are sent.
- ✓ The interval of time, in seconds, after which a route is declared invalid.
- ✓ The interval, in seconds, during which routing information about better paths is suppressed.
- ✓ The amount of time, in seconds, that must pass before a route is removed from the routing table.
- ✓ The amount of time for which routing updates will be postponed.

SIMULATION OF RIP PROTOCOL

Step 1: Make the topology with two routers, two switches and two client computers as below as tools available in packet tracer.



Step 2: Double click the PC0 – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.1.2** – subnet 255.255.255.0 – default gateway – 192.168.1.1.


PC0

IP Configuration

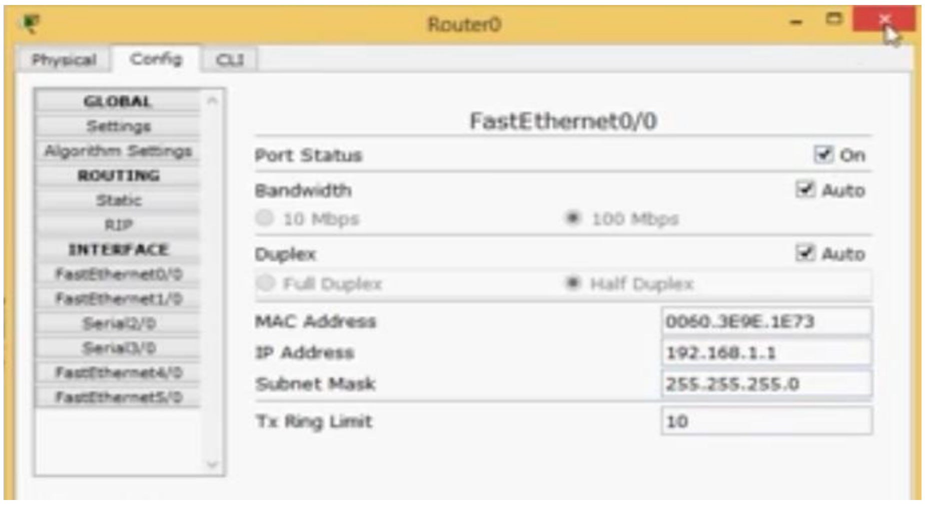
IP Configuration

DHCP [✓ Static]

IP Address	192.168.1.2
Subnet Mask	255.255.255.0
Default Gateway	192.168.1.1
DNS Server	

Step 3: Similarly Double click the PC1 – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.2.2** – subnet 255.255.255.0 – default gateway – 192.168.2.1.

Step 4: Double click the Router0 – Config – choose FastEthernet0/0 – make sure check Boxes port status : on, bandwidth: auto, duplex:auto are checked - enter ip address as **192.168.1.1** – subnet 255.255.255.0.



FastEthernet0/0

Port Status ☑ On

Bandwidth ☑ Auto

○ 10 Mbps ● 100 Mbps

Duplex ☑ Auto

○ Full Duplex ● Half Duplex

MAC Address 0060.3E9E.1E73

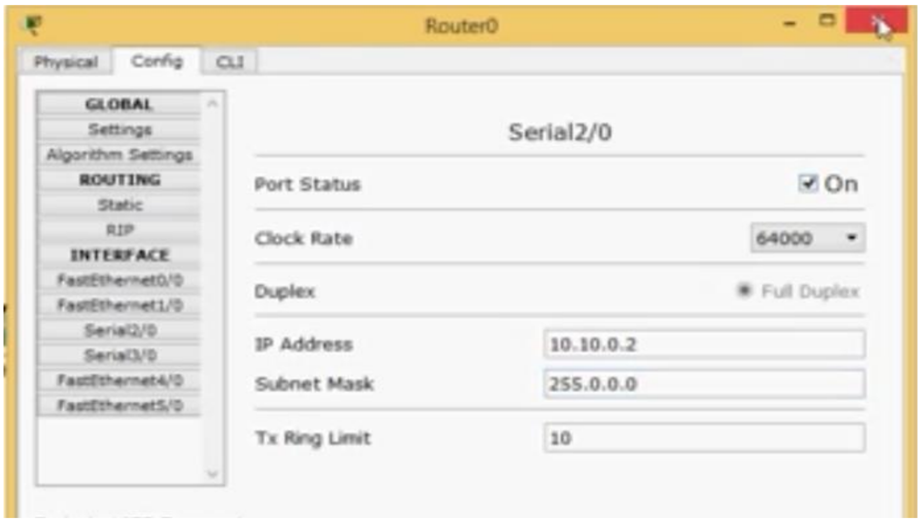
IP Address 192.168.1.1

Subnet Mask 255.255.255.0

Tx Ring Limit 10

Step 5: Similarly Double click the Router1 – Config – choose FastEthernet0/0 – make sure check Boxes port status : on, bandwidth: auto, duplex: auto are checked - enter ip address as **192.168.2.1** – subnet 255.255.255.0.

Step 6: Similarly Double click the Router0 – Config – choose Serial2/0 – make sure check Boxes port status : "on" is checked, set clock rate is : 64000, enable radio button on duplex: full duplex - enter ip address as **10.10.0.2** – subnet 255.0.0.0.



Serial2/0

Port Status ☑ On

Clock Rate 64000

Duplex ● Full Duplex

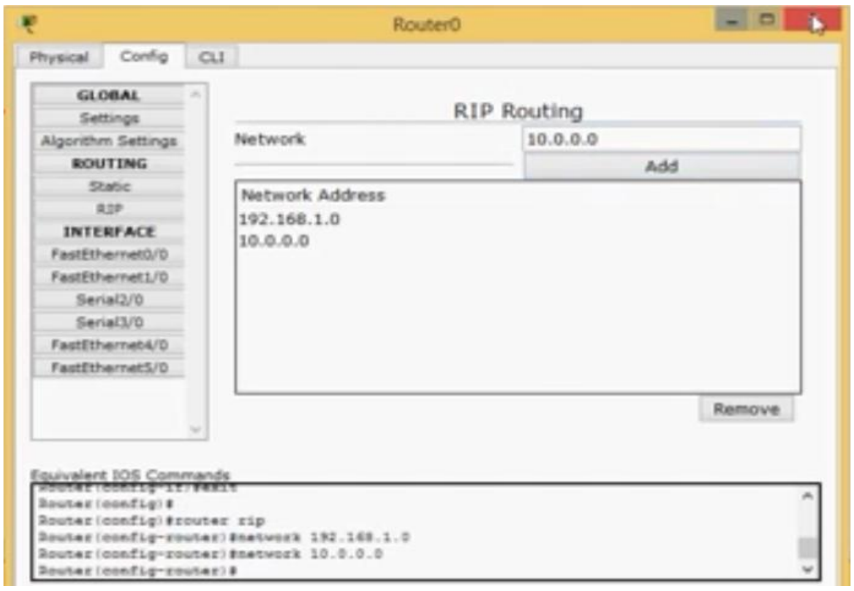
IP Address 10.10.0.2

Subnet Mask 255.0.0.0

Tx Ring Limit 10

Step 7: Similarly Double click the Router1 – Config – choose Serial2/0 – make sure check Boxes port status : "on" is checked, set clock rate is : 64000, enable radio button on duplex: full duplex - enter ip address as **10.10.0.2** – subnet 255.0.0.0.

Step 8: Double click the Router0 – Config – choose RIP – add the following ip address 192.168.1.0 and 10.0.0.0 using "Add" button – choose settings – click "save" button. Administrator can do this process also using ios commands.



Network 10.0.0.0

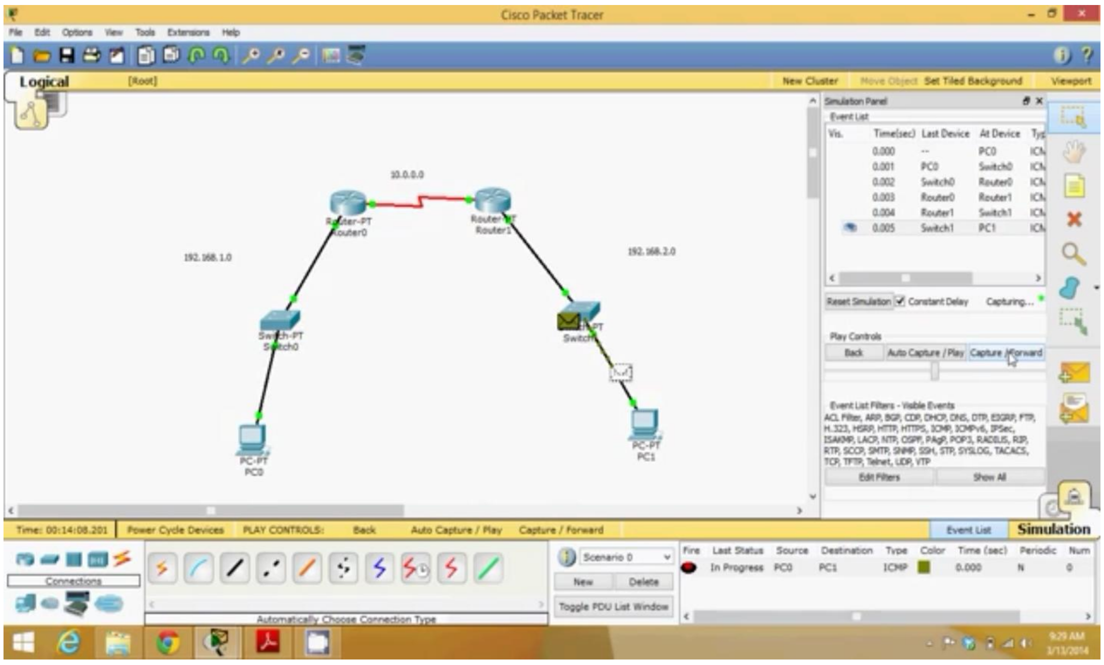
Network Address
192.168.1.0
10.0.0.0

Equivalent IOS Commands

```
Router(config-if)#exit
Router(config)#
Router(config)#router rip
Router(config-router)#network 192.168.1.0
Router(config-router)#network 10.0.0.0
Router(config-router)#
```

Step 9: Simiarly Double click the Router0 – Config – choose RIP – add the following ip address 192.168.1.0 and 10.0.0.0 using "Add" button – choose settings – click "save" button. Administrator can do this process also using ios commands.

Step 10: By using the sample test packets from the options available, evaluating the performance RIP based routing network configuration in simulation mode as per the above information about routing information protocol is occurred successfully.



Vis.	Time(sec)	Last Device	At Device	Typ
	0.000	--	PC0	ICM
	0.001	PC0	Switch0	ICM
	0.002	Switch0	Router0	ICM
	0.003	Router0	Router1	ICM
	0.004	Router1	Switch1	ICM
	0.005	Switch1	PC1	ICM

Event List Filters - Visible Events
ACL Filter, ARP, BGP, CDP, DHCP, DNS, DTP, EIGRP, FTP, H.323, HSRP, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NTP, OSPF, PAgP, POP3, RADIUS, RIP, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Num
●	In Progress	PC0	PC1	ICMP	🟩	0.000	N	0

OSPF – OPEN SHORTEST PATH FIRST PROTOCOL

OSPF is a standardized Link-State routing protocol, designed to scale efficiently to support larger networks.

OSPF adheres to the following Link State characteristics:

- ✓ OSPF employs a hierarchical network design using Areas.
- ✓ OSPF will form neighbour relationships with adjacent routers in the same Area.
- ✓ Instead of advertising the distance to connected networks, OSPF advertises the status of directly connected links using Link-State Advertisements (LSAs).
- ✓ OSPF sends updates (LSAs) when there is a change to one of its links, and will only send the change in the update. LSAs are additionally refreshed every 30 minutes.
- ✓ OSPF traffic is multicast either to address 224.0.0.5 (all OSPF routers) or 224.0.0.6(all Designated Routers).
- ✓ OSPF uses the Dijkstra Shortest Path First algorithm to determine the shortest path.
- ✓ OSPF is a classless protocol, and thus supports VLSMs.

Other characteristics of OSPF include:

- ✓ OSPF supports only IP routing.
- ✓ OSPF routes have an administrative distance is 110.
- ✓ OSPF uses cost as its metric, which is computed based on the bandwidth of the link.
- ✓ OSPF has no hop-count limit.

The OSPF process builds and maintains three separate tables:

- ✓ A neighbour table – contains a list of all neighbouring routers.
- ✓ A topology table– contains a list of all possible routes to all known networks within an area.
- ✓ A routing table– contains the best route for each known network.

OSPF forms neighbour relationships, called adjacencies, with other routers in the same Area by exchanging Hello packets to multicast address 224.0.0.5. Only after an adjacency is formed can routers share routing information.

Each OSPF router is identified by a unique Router ID. The Router ID can be determined in one of three ways:

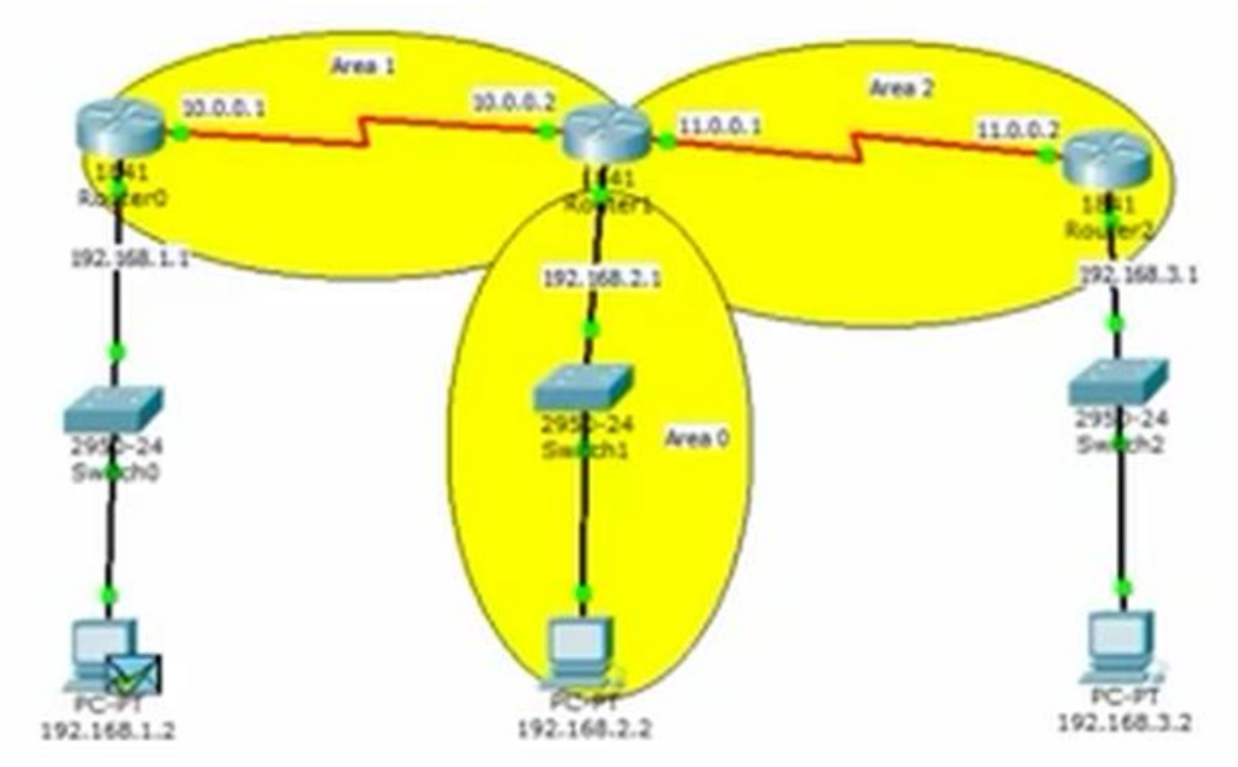
- ✓ The Router ID can be manually specified.
- ✓ If not manually specified, the highest IP address configured on any
- ✓ Loopback interface on the router will become the Router ID.
- ✓ If no loopback interface exists, the highest IP address configured on any Physical interface will become the Router ID.

By default, Hello packets are sent out OSPF-enabled interfaces every 10 seconds for broadcast and point-to-point interfaces, and 30 seconds for non-broadcast and point-to-multipoint interfaces.

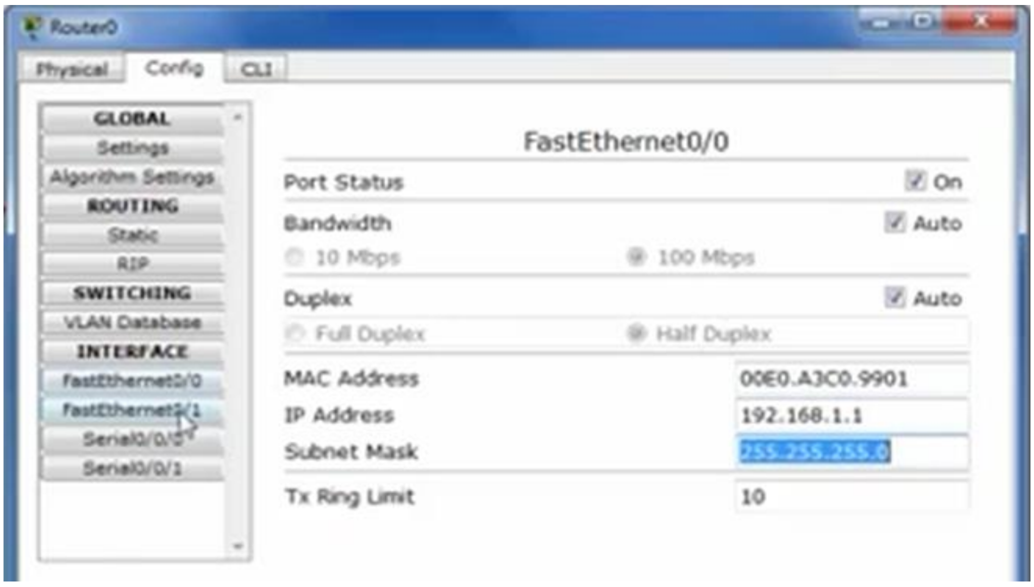
OSPF also has a Dead Interval, which indicates how long a router will wait without hearing any hellos before announcing a neighbour as “down.” Default for the Dead Interval is 40 seconds for broadcast and point-to-point interfaces, and 120seconds for non-broadcast and point-to-multipoint interfaces. Notice that, by default, the dead interval timer is four times the Hello interval.

SIMULATION OF OSPF PROTOCOL

Step 1: Make the topology with two routers, two switches and two client computers as below as tools available in packet tracer.



Step 2: Double click the Router0 – Config – choose FastEthernet0/0 – make sure check Boxes port status : on, bandwidth: auto, duplex: auto are checked - enter ip address as **192.168.1.1** – subnet 255.255.255.0.



FastEthernet0/0

Port Status ☑ On

Bandwidth ☑ Auto

○ 10 Mbps ● 100 Mbps

Duplex ☑ Auto

○ Full Duplex ● Half Duplex

MAC Address 00E0.A3C0.9901

IP Address 192.168.1.1

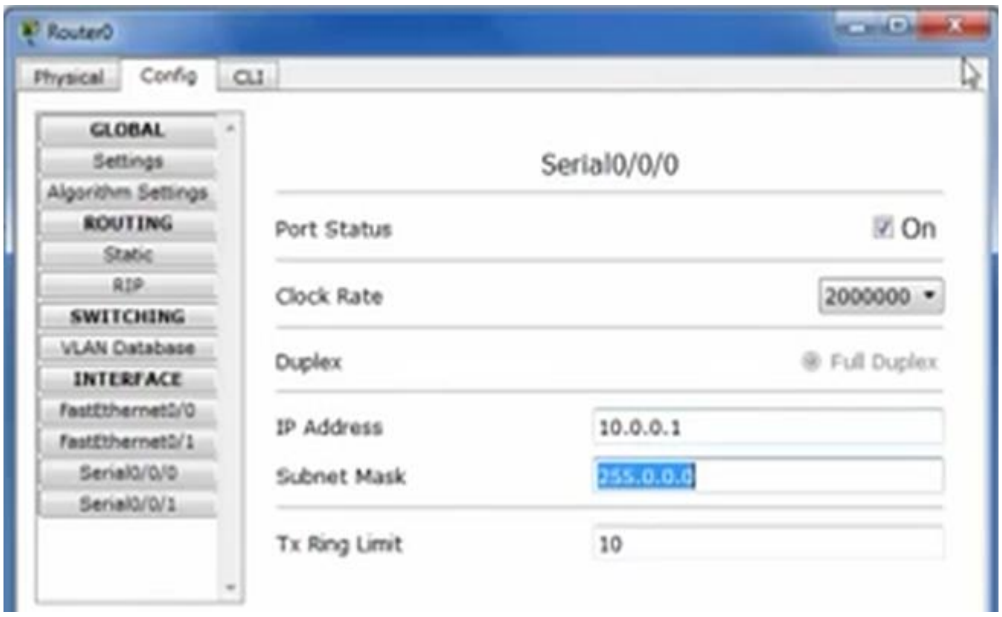
Subnet Mask 255.255.255.0

Tx Ring Limit 10

Step 3: Similarly Double click the Router1 – Config – choose FastEthernet0/0 – make sure check Boxes port status : on, bandwidth: auto, duplex: auto are checked - enter ip address as **192.168.2.1** – subnet 255.255.255.0.

Step 4: Similarly Double click the Router2 – Config – choose FastEthernet0/0 – make sure check Boxes port status : on, bandwidth: auto, duplex: auto are checked - enter ip address as **192.168.3.1** – subnet 255.255.255.0.

Step 5: Double click the Router0 – Config – choose Serial0/0/0 – make sure check Boxes port status : "on" is checked, set clock rate is : 200000, enable radio button on duplex: full duplex - enter ip address as **10.0.0.1** – subnet 255.0.0.0.



Serial0/0/0

Port Status ☑ On

Clock Rate 2000000

Duplex ● Full Duplex

IP Address 10.0.0.1

Subnet Mask 255.0.0.0

Tx Ring Limit 10

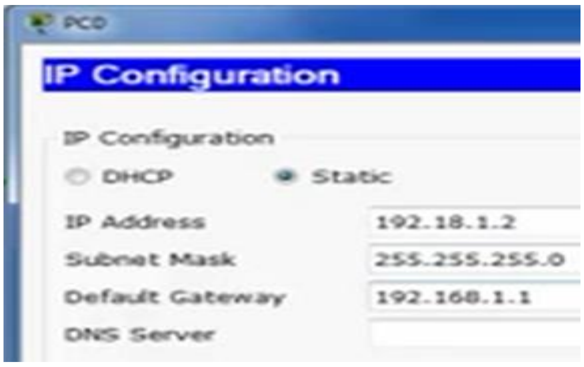
Step 6: Double click the Router1 – Config – choose Serial0/0/0 – make sure check Boxes port status : "on" is checked, set clock rate is : 200000, enable radio button on duplex: full duplex - enter ip address as **10.0.0.2** – subnet 255.0.0.0.

Step 7: Double click the Router1 – Config – choose Serial0/0/1 – make sure check Boxes port status : "on" is checked, set clock rate is : 200000, enable radio button on duplex: full duplex - enter ip address as **11.0.0.1** – subnet 255.0.0.0.

Step 8: Double click the Router2 – Config – choose Serial0/0/0 – make sure check Boxes port status : "on" is checked, set clock rate is : 200000, enable radio button on duplex: full duplex - enter ip address as **11.0.0.2** – subnet 255.0.0.0.

Step 9: Double click the PC0 – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.1.2** – subnet 255.255.255.0 – default gateway – 192.168.1.1.

Step 10: Similarly Double click the PC1 – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.2.2** – subnet 255.255.255.0 – default gateway – 192.168.2.1.


PC0

IP Configuration

IP Configuration

DHCP [✓ Static]

IP Address 192.18.1.2

Subnet Mask 255.255.255.0

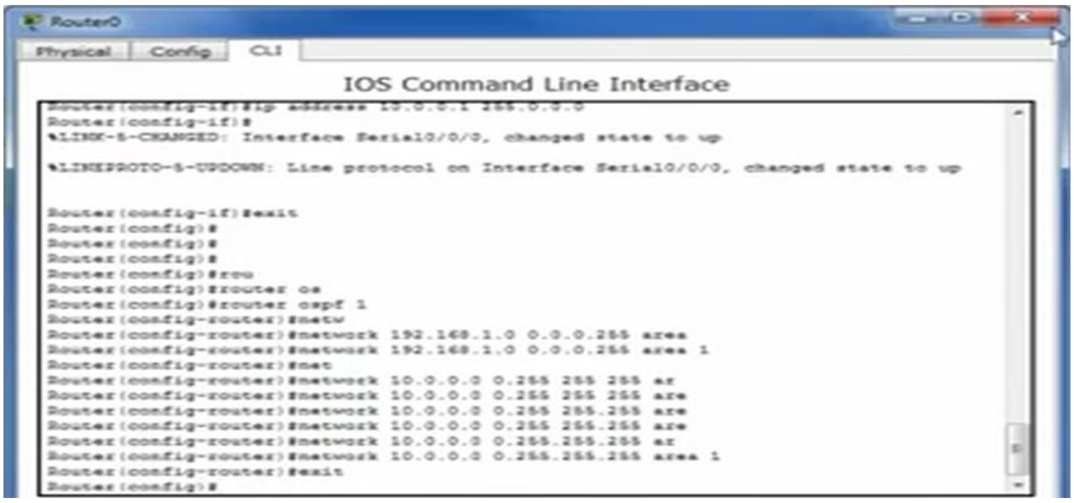
Default Gateway 192.168.1.1

DNS Server

Step 10: Similarly Double click the PC2 – desktop – choose ip configuration – make sure enable radio button on static- enter ip address as **192.168.3.2** – subnet 255.255.255.0 – default gateway – 192.168.3.1.

Step 11: Double click the Router0 – CLI – enter the following ios commands in IOS Command Line Interface to configure OSPF routing network on router0.

```
Router (config-if)#exit
Router (config)#
Router (config)#
Router (config)#
Router (config)#rou
Router (config)#router os
Router (config)#router ospf 1
Router (config-router)#netw
Router (config-router)#network 192.168.1.0 0.0.0.255 area
Router (config-router)#network 192.168.1.0 0.0.0.255 area 1
Router (config-router)#net
Router (config-router)#network 10.0.0.0 0.255.255.255 ar
Router (config-router)#network 10.0.0.0 0.255.255.255 are
Router (config-router)#network 10.0.0.0 0.255.255.255 are
Router (config-router)#network 10.0.0.0 0.255.255.255 are
Router (config-router)#network 10.0.0.0 0.255.255.255 ar
Router (config-router)#network 10.0.0.0 0.255.255.255 area 1
Router (config-router)#exit
Router (config)#
```



```
Router(config-if)#ip address 10.0.0.1 255.0.0.0
Router(config-if)#
%LINK-5-CHANGED: Interface Serial0/0/0, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0/0, changed state to up


Router(config-if)#exit
Router(config)#
Router(config)#
Router(config)#
Router(config)#rou
Router(config)#router os
Router(config)#router ospf 1
Router(config-router)#netw
Router(config-router)#network 192.168.1.0 0.0.0.255 area
Router(config-router)#network 192.168.1.0 0.0.0.255 area 1
Router(config-router)#net
Router(config-router)#network 10.0.0.0 0.255 255 255 ar
Router(config-router)#network 10.0.0.0 0.255 255 255 are
Router(config-router)#network 10.0.0.0 0.255 255.255 are
Router(config-router)#network 10.0.0.0 0.255 255.255 are
Router(config-router)#network 10.0.0.0 0.255.255.255 ar
Router(config-router)#network 10.0.0.0 0.255.255.255 area 1
Router(config-router)#exit
Router(config)#
```

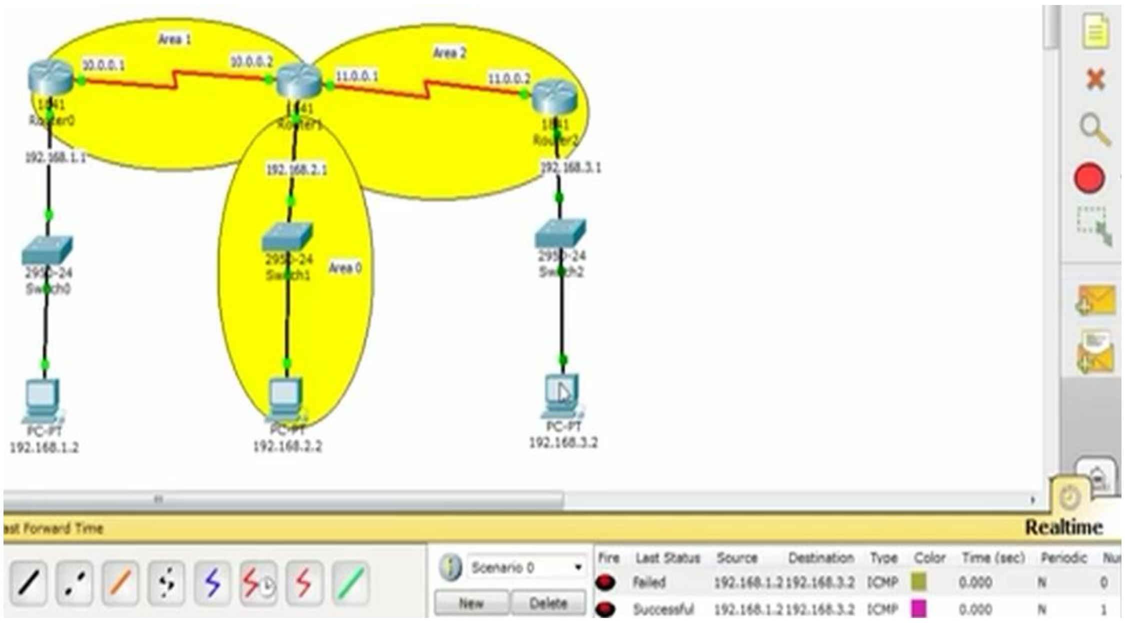
Step 12: Similarly Double click the Router1 – CLI – enter the following ios commands in IOS Command Line Interface to configure OSPF routing network on router1.

```
Router (config-if)#exit
Router (config)#rout
Router (config)#router osp
Router (config)#router ospf 2
Router (config-router)#net
Router (config-router)#network 192.168.2.0 0.0.0.255 ar
Router (config-router)#network 192.168.1.0 0.0.0.255 area 0
Router (config-router)#net
Router (config-router)#network 10.0.0.0 0.255.255.255 are
Router (config-router)#network 10.0.0.0 0.255.255.255 area 1
Router (config-router)#net
Router (config-router)#network
Router (config-router)#net
Router (config-router)#network 11.0.0.0 0.255.255.255 ar
Router (config-router)#network 11.0.0.0 0.255.255.255 area 2
Router (config-router)#exit
Router (config)#
```

Step 12: Similarly Double click the Router2 – CLI – enter the following ios commands in IOS Command Line Interface to configure OSPF routing network on router2.

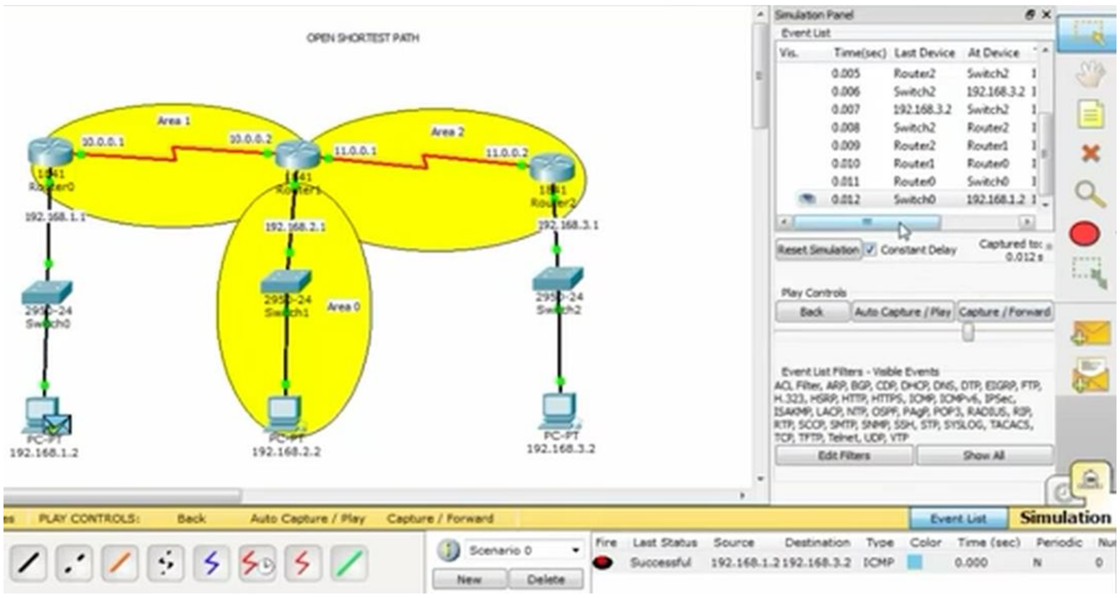
```
Router (config-if)#exit
Router (config)#rou
Router (config)#router os
Router (config)#router ospf 1
Router (config-router)#net
Router (config-router)#network 192.168.2.0 0.0.0.255 ar
Router (config-router)#network 192.168.1.0 0.0.0.255 area 2
Router (config-router)#net
Router (config-router)#network 10.0.0.0 0.255.255.255 are
Router (config-router)#network 10.0.0.0 0.255.255.255 area 2
Router (config-router)exit
Router (config)#
```

Step 12: Test OSPF network configuration using sample packets in real time mode from PC0 (area 0) to PC2 (area 2).



Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Nu
●	Failed	192.168.1.2	192.168.3.2	ICMP	🟩	0.000	N	0
●	Successful	192.168.1.2	192.168.3.2	ICMP	🟪	0.000	N	1

Step 13: Test OSPF network configuration using sample packets in simulation mode from PC0 (area 0) to PC2 (area 2). Analyse the successful packet sending and receiving between different networks.



Event List

Vis.	Time(sec)	Last Device	At Device	
	0.005	Router2	Switch2	1
	0.006	Switch2	192.168.3.2	1
	0.007	192.168.3.2	Switch2	1
	0.008	Switch2	Router2	1
	0.009	Router2	Router1	1
	0.010	Router1	Router0	1
	0.011	Router0	Switch0	1
	0.012	Switch0	192.168.1.2	1

Constant Delay [✓]

Captured to: 0.012 s

Event List Filters - Visible Events

ACL Filter, ARP, BGP, CDP, DHCP, DNS, DTP, EIGRP, FTP, H.323, HSRP, HTTP, HTTPS, ICMP, ICMPv6, IPSec, ISAKMP, LACP, NTP, OSPF, PAgP, POP3, RADIUS, RIP, RTP, SCCP, SMTP, SNMP, SSH, STP, SYSLOG, TACACS, TCP, TFTP, Telnet, UDP, VTP

Fire	Last Status	Source	Destination	Type	Color	Time (sec)	Periodic	Nu
●	Successful	192.168.1.2	192.168.3.2	ICMP	🟦	0.000	N	0

Step 12: Double click the Router0 – CLI – to check shortest network enter the following ios commands in IOS Command Line Interface on router2.

Router (config)#

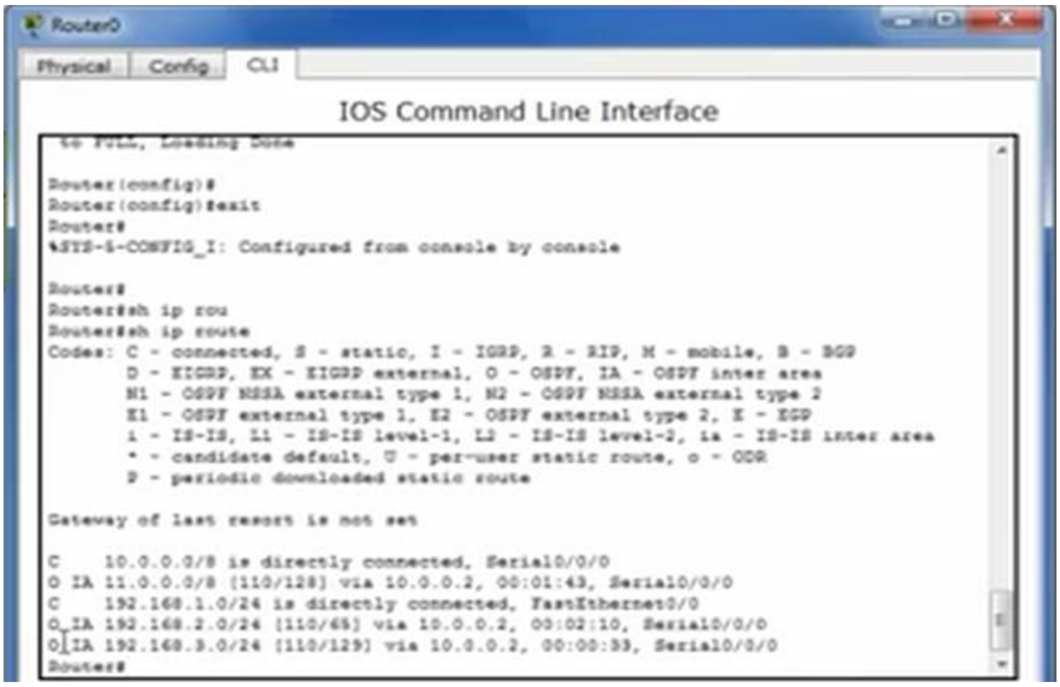
Router (config)#exit

Router#

Router#sh ip rou

Router#sh ip route\

Similarly do this process in router1 and router2 to know about shortest network path selecting of a particular network.



```
 to FULL, Loading Done

Router(config)#
Router(config)#exit
Router#
%SYS-5-CONFIG_I: Configured from console by console

Router#
Router#sh ip rou
Router#sh 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, Serial0/0/0
O IA 11.0.0.0/8 [110/128] via 10.0.0.2, 00:01:43, Serial0/0/0
C    192.168.1.0/24 is directly connected, FastEthernet0/0
O IA 192.168.2.0/24 [110/65] via 10.0.0.2, 00:02:10, Serial0/0/0
O IA 192.168.3.0/24 [110/129] via 10.0.0.2, 00:00:33, Serial0/0/0
Router#
```

RESULT :

The analysis of performance evaluation of routing protocols using simulation tool has been done successfully.

Ex.No:10 **Simulation of Error Detection Code (like CRC)**

DATE :

AIM:

To write a java program fot simulating of error correction code

ALGORITHM:

1. Start the Program
2. Given a bit string, append 0S to the end of it (the number of 0s is the same as the degree of the generator polynomial) let B(x) be the polynomial corresponding to B.
3. Divide B(x) by some agreed on polynomial G(x) (generator polynomial) and determinethe remainder R(x). This division is to be done using Modulo 2 Division.
4. Define T(x) = B(x) –R(x)
5. (T(x)/G(x) => remainder 0)
6. Transmit T, the bit string corresponding to T(x).
7. Let T’ represent the bit stream the receiver gets and T’(x) the associated polynomial. The receiver divides T1(x) by G(x). If there is a 0 remainder, the receiver concludes T = T’ and no error occurred otherwise, the receiver concludes an error occurred and requires a retransmission
8. Stop the Program.

PROGRAM :

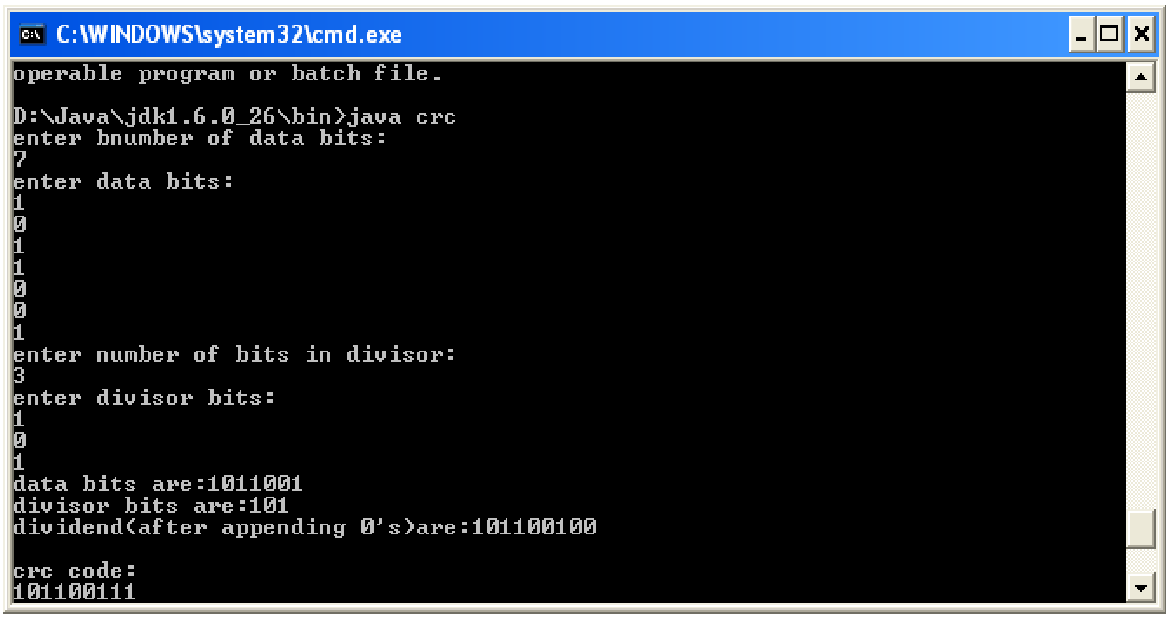
```
import java.io.*;
class crc
{
public static void main(String args[])throws IOException
{
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
int[] data;
int[]div;
int[]divisor;
int[]rem;
int[]crc;
int data_bits,divisor_bits,tot_length;
System.out.println("enter bnumber of data bits:");
data_bits=Integer.parseInt(br.readLine());
data=new int[data_bits];
System.out.println("enter data bits:");
for(int i=0;i<data_bits;i++)
data[i]=Integer.parseInt(br.readLine());
System.out.println("enter number of bits in divisor:");
divisor_bits=Integer.parseInt(br.readLine());
divisor=new int[divisor_bits];
System.out.println("enter divisor bits:");
for(int i=0;i<divisor_bits;i++)
divisor[i]=Integer.parseInt(br.readLine());
System.out.print("data bits are:");
for(int i=0; i<data_bits;i++)
```

```
System.out.print(data[i]);
System.out.println();
System.out.print("divisor bits are:");
for(int i=0;i<divisor_bits;i++)
System.out.print(divisor[i]);
System.out.println();
tot_length=data_bits+divisor_bits-1;
div=new int[tot_length];
rem=new int[tot_length];
crc=new int[tot_length];
for(int i=0;i<data.length;i++)
div[i]=data[i];
System.out.print("dividend(after appending 0's)are:");
for(int i=0;i<div.length;i++)
System.out.print(div[i]);
System.out.println();
for(int j=0;j<div.length;j++){
rem[j]=div[j];
}
rem=divide(div,divisor,rem);
for(int i=0;i<div.length;i++)
{
crc[i]=(div[i]^rem[i]);
}
System.out.println();
System.out.println("crc code:");
for(int i=0;i<crc.length;i++)System.out.print(crc[i]);
```

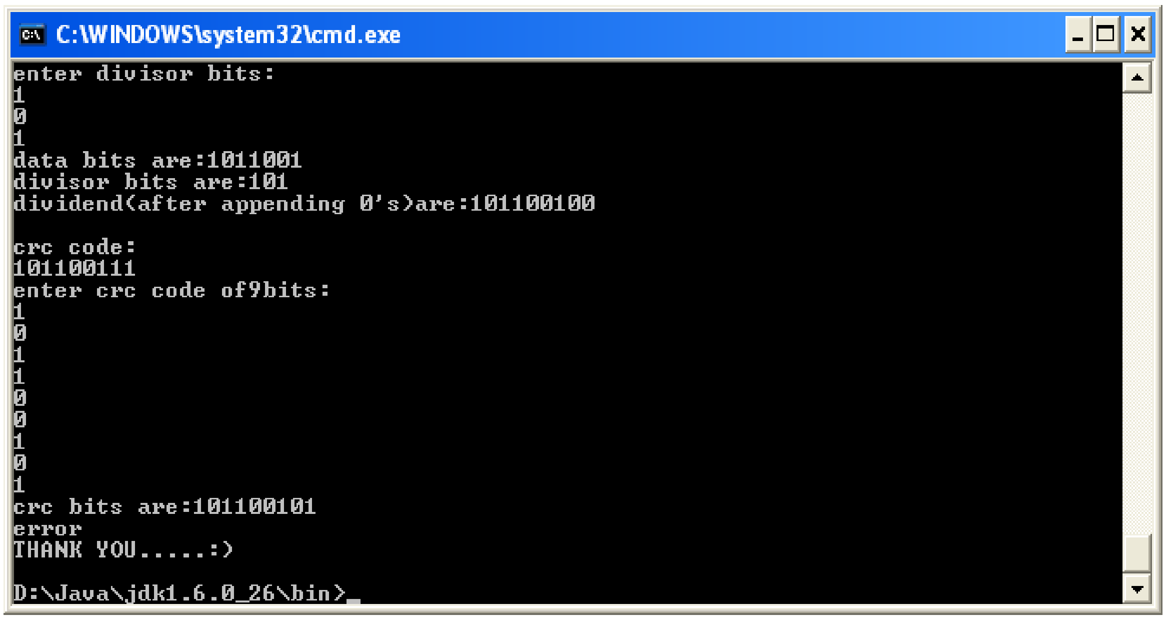
```
System.out.println();
System.out.println("enter crc code of"+tot_length+"bits:");
for(int i=0;i<crc.length;i++)
crc[i]=Integer.parseInt(br.readLine());
System.out.print("crc bits are:");
for(int i=0;i<crc.length;i++)
System.out.print(crc[i]);
System.out.println();
for(int j=0;j<crc.length;j++){
rem[j]=crc[j];
}
rem=divide(crc,divisor,rem);
for(int i=0;i<rem.length;i++)
{
if(rem[i]!=0)
{
System.out.println("error");
break;
}
if(i==rem.length-1)
System.out.println("No error");
}
System.out.println("THANK YOU.....:)");
}
static int[] divide(int div[],int divisor[],int rem[])
{
int cur=0;
```

```
while(true)
{
for(int i=0;i<divisor.length;i++)
rem[cur+i]=(rem[cur+i]^divisor[i]);
while(rem[cur]==0&&cur!=rem.length-1)
cur++;
if((rem.length-cur)<divisor.length)
break;
}
return rem;
}
}
```

OUTPUT :



```
operable program or batch file.

D:\Java\jdk1.6.0_26\bin>java crc
enter bnumber of data bits:
7
enter data bits:
1
0
1
1
0
0
1
enter number of bits in divisor:
3
enter divisor bits:
1
0
1
data bits are:1011001
divisor bits are:101
dividend(after appending 0's)are:101100100

crc code:
101100111
```



```
enter divisor bits:
1
0
1
data bits are:1011001
divisor bits are:101
dividend(after appending 0's)are:101100100

crc code:
101100111
enter crc code of9bits:
1
0
1
1
0
0
1
0
1
crc bits are:101100101
error
THANK YOU.....:)

D:\Java\jdk1.6.0_26\bin>_
```

RESULT:

Thus the program was execute successfully .