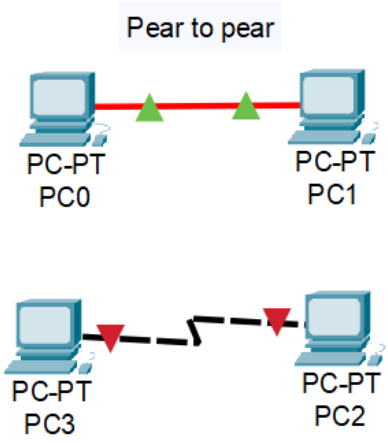
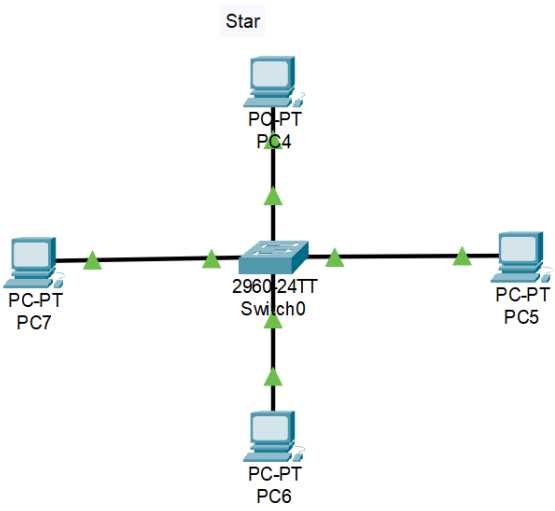
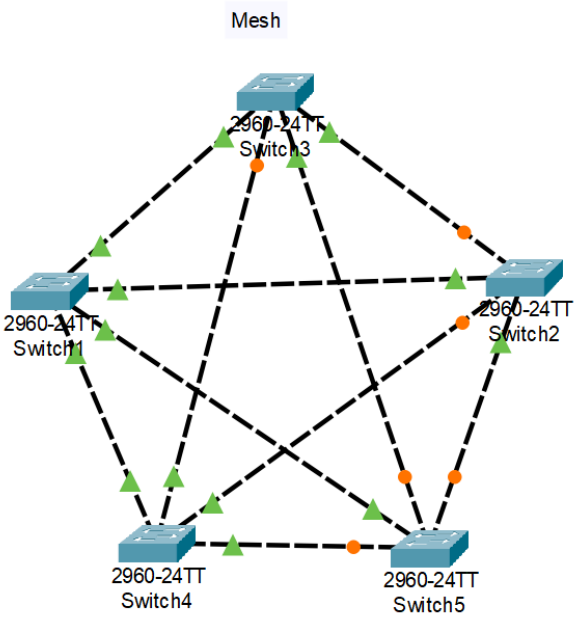
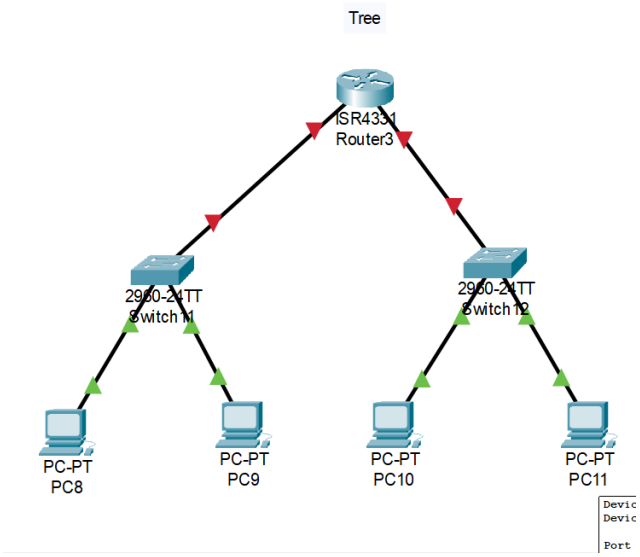
Practical No. 01

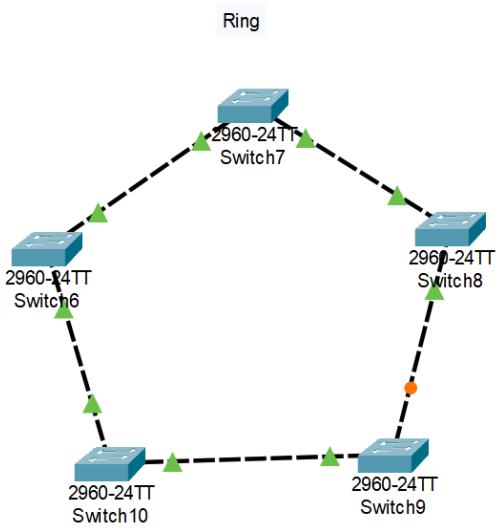
Demonstrate the different types of topologies and types of transmission media by using a packet tracer tool.

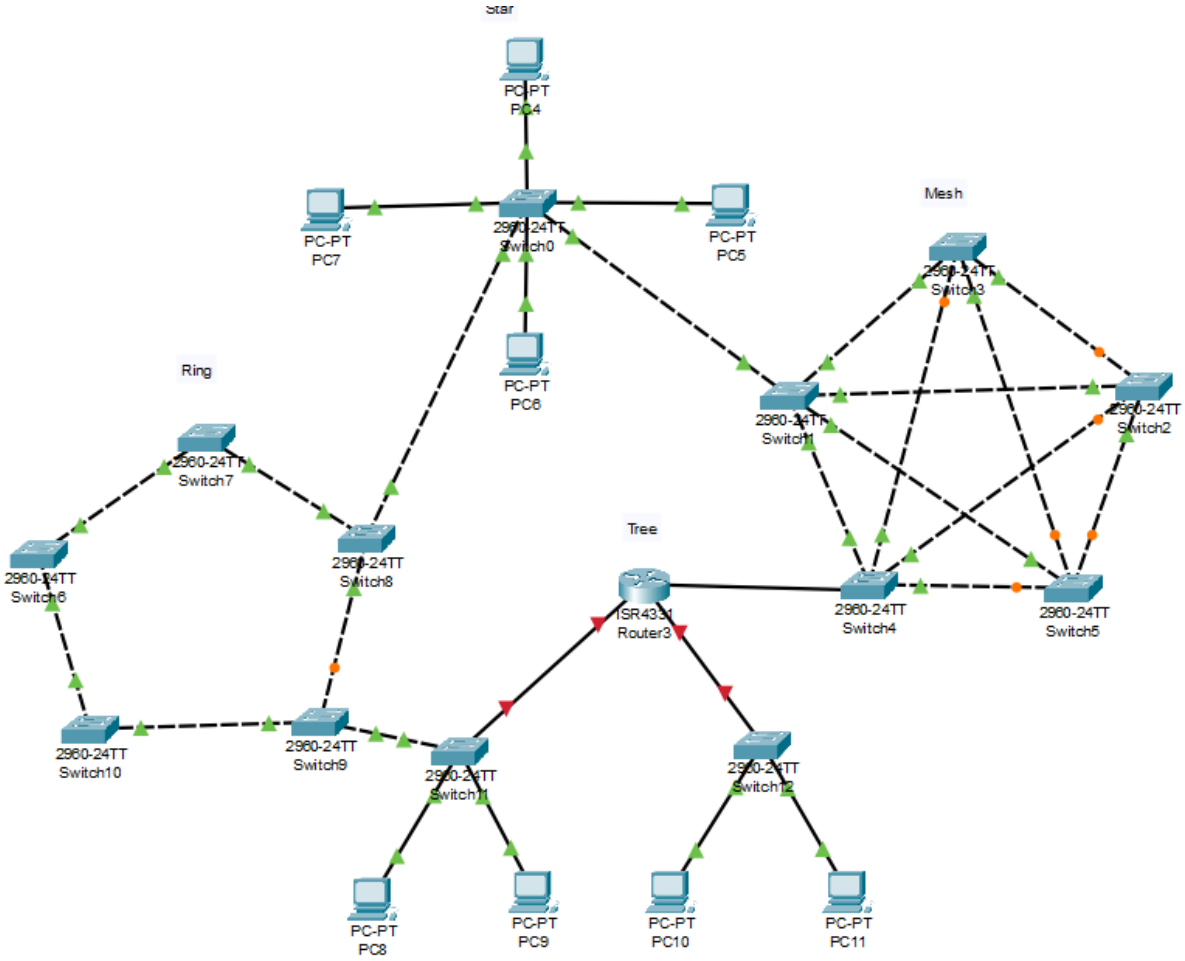









Ring
2960-24TT
Switch7
2960-24TT
Switch6
2960-24TT
Switch8
2960-24TT
Switch10
2960-24TT
Switch9


Star
PC-PT
PC4
PC-PT
PC7
2960-24TT
Switch0
PC-PT
PC5
PC-PT
PC6
Mesh
2960-24TT
Switch3
2960-24TT
Switch1
2960-24TT
Switch2
Ring
2960-24TT
Switch7
2960-24TT
Switch6
2960-24TT
Switch8
Tree
ISR4331
Router3
2960-24TT
Switch4
2960-24TT
Switch5
2960-24TT
Switch10
2960-24TT
Switch9
2960-24TT
Switch11
2960-24TT
Switch12
PC-PT
PC8
PC-PT
PC9
PC-PT
PC10
PC-PT
PC11

Practical No. 02

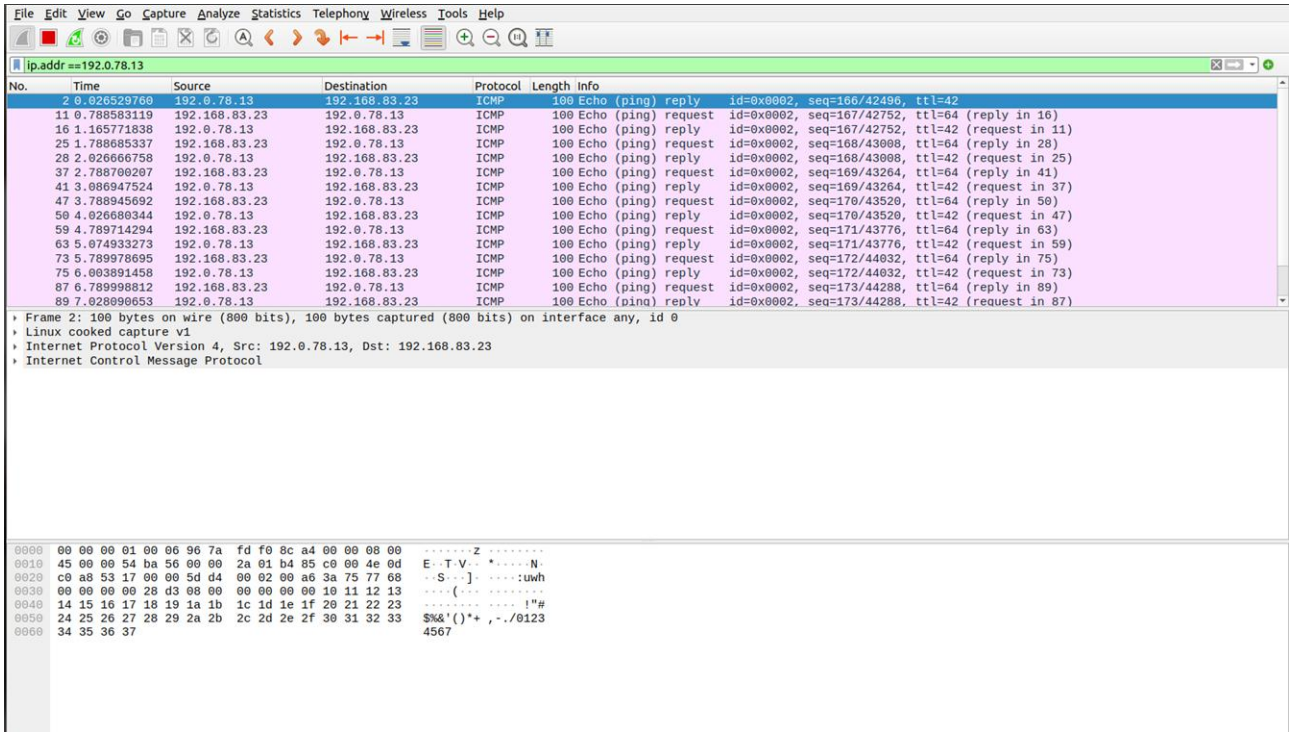
Setup a wired LAN using Layer 2 Switch. It includes preparation of cable, testing of cable using line tester, configuration machine using IP addresses, testing using PING utility and demonstrating the PING packets captured traces using Wireshark Packet Analyzer Tool.

```
(base) aiml@aiml-ThinkCentre-M900:~$ ifconfig
eno1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.16.0.122  netmask 255.255.0.0  broadcast 172.16.255.255
        inet6 fe80::7333:6945:cc3a:9c8a  prefixlen 64  scopeid 0x20<link>
        ether d8:cb:8a:d4:f8:9e  txqueuelen 1000  (Ethernet)
        RX packets 45841  bytes 5081543 (5.0 MB)
        RX errors 0  dropped 230  overruns 0  frame 0
        TX packets 2274  bytes 356980 (356.9 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 16  memory 0xdf000000-df020000

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 3346  bytes 282241 (282.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3346  bytes 282241 (282.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

(base) aiml@aiml-ThinkCentre-M900:~$
```

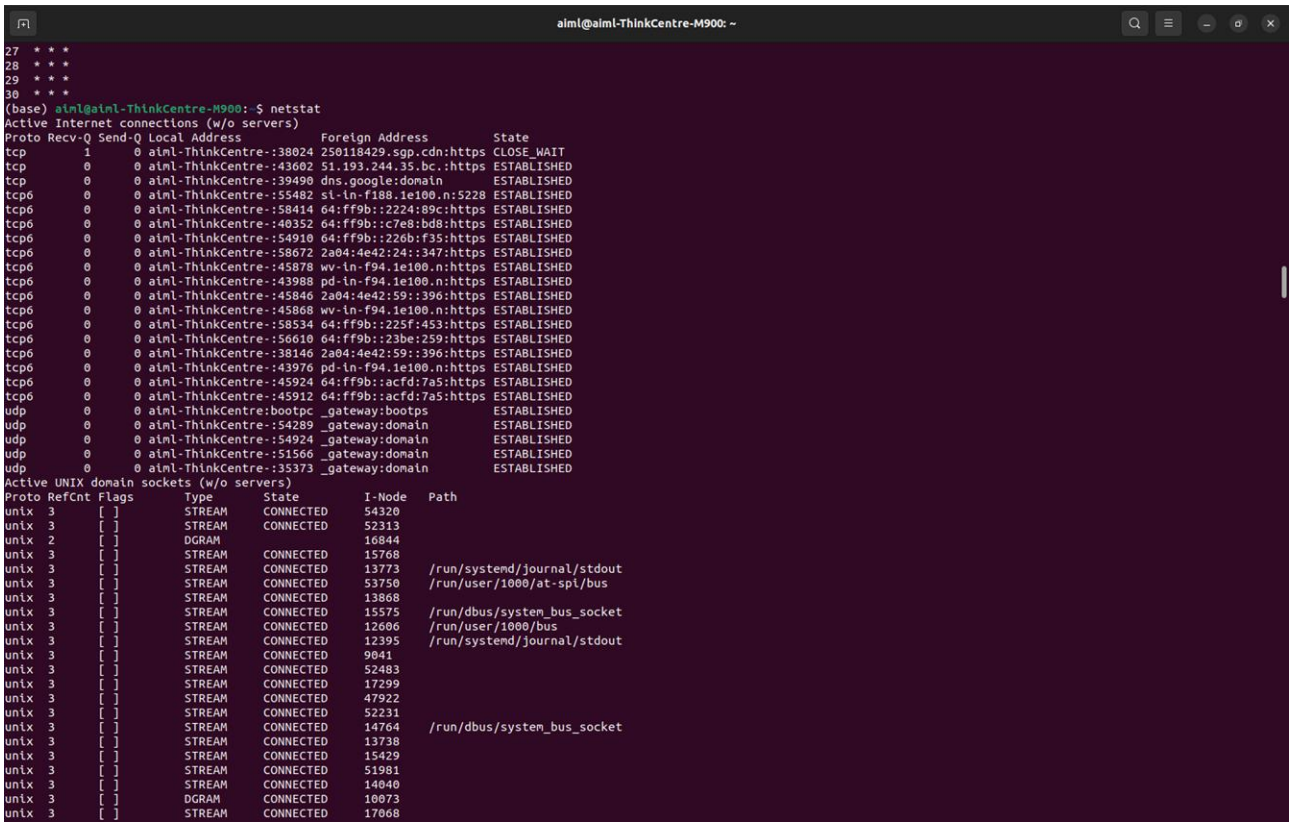
```
(base) aiml@aiml-ThinkCentre-M900:~$ hostname
aiml-ThinkCentre-M900
(base) aiml@aiml-ThinkCentre-M900:~$
```

```
(base) aiml@aiml-ThinkCentre-M900:~$ nslookup
> www.abphotovideographics.wordpress.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
www.abphotovideographics.wordpress.com  canonical name = lb.wordpress.com.
Name:   lb.wordpress.com
Address: 192.0.78.13
Name:   lb.wordpress.com
Address: 192.0.78.12
Name:   lb.wordpress.com
Address: 64:ff9b::c000:4e0c
Name:   lb.wordpress.com
Address: 64:ff9b::c000:4e0d
>
```

```
(base) aiml@aiml-ThinkCentre-M900:~$ ping 192.0.78.13
PING 192.0.78.13 (192.0.78.13) 56(84) bytes of data.
64 bytes from 192.0.78.13: icmp_seq=1 ttl=42 time=240 ms
64 bytes from 192.0.78.13: icmp_seq=2 ttl=42 time=237 ms
64 bytes from 192.0.78.13: icmp_seq=3 ttl=42 time=271 ms
64 bytes from 192.0.78.13: icmp_seq=4 ttl=42 time=336 ms
64 bytes from 192.0.78.13: icmp_seq=5 ttl=42 time=257 ms
64 bytes from 192.0.78.13: icmp_seq=6 ttl=42 time=235 ms
64 bytes from 192.0.78.13: icmp_seq=7 ttl=42 time=238 ms
64 bytes from 192.0.78.13: icmp_seq=8 ttl=42 time=237 ms
```



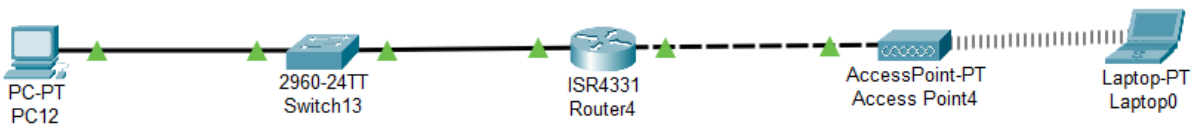

```
(base) aiml@aiml-ThinkCentre-M900:~$ traceroute 192.0.78.13
traceroute to 192.0.78.13 (192.0.78.13), 30 hops max, 60 byte packets
 1  _gateway (192.168.83.5)  1.445 ms  1.644 ms  1.850 ms
 2  10.229.255.254 (10.229.255.254)  62.085 ms * *
 3  * * *
 4  * * *
 5  * * *
 6  192.168.100.5 (192.168.100.5)  68.648 ms  23.939 ms  35.485 ms
 7  * * *
 8  10.174.169.65 (10.174.169.65)  47.696 ms  22.499 ms  32.006 ms
 9  118.185.22.90 (118.185.22.90)  27.365 ms  33.967 ms  30.559 ms
```


aiml@aiml-ThinkCentre-M900: ~
27 * * *
28 * * *
29 * * *
30 * * *
(base) aiml@aiml-ThinkCentre-M900:~$ netstat
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp 1 0 aiml-ThinkCentre-:38024 250118429.sgp.cdn:https CLOSE_WAIT
tcp 0 0 aiml-ThinkCentre-:43602 51.193.244.35.bc.:https ESTABLISHED
tcp 0 0 aiml-ThinkCentre-:39490 dns.google:domain ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:55482 si-in-f188.1e100.n:5228 ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:58414 64:ff9b::2224:89c:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:40352 64:ff9b::c7e8:bd8:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:54910 64:ff9b::226b:f35:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:58672 2a04:4e42:24::347:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:45878 wv-in-f94.1e100.n:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:43988 pd-in-f94.1e100.n:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:45846 2a04:4e42:59::396:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:45868 wv-in-f94.1e100.n:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:58534 64:ff9b::225f:453:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:56610 64:ff9b::23be:259:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:38146 2a04:4e42:59::396:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:43976 pd-in-f94.1e100.n:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:45924 64:ff9b::acfd:7a5:https ESTABLISHED
tcp6 0 0 aiml-ThinkCentre-:45912 64:ff9b::acfd:7a5:https ESTABLISHED
udp 0 0 aiml-ThinkCentre:bootpc _gateway:bootps ESTABLISHED
udp 0 0 aiml-ThinkCentre-:54289 _gateway:domain ESTABLISHED
udp 0 0 aiml-ThinkCentre-:54924 _gateway:domain ESTABLISHED
udp 0 0 aiml-ThinkCentre-:51566 _gateway:domain ESTABLISHED
udp 0 0 aiml-ThinkCentre-:35373 _gateway:domain ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags Type State I-Node Path
unix 3 [ ] STREAM CONNECTED 54320
unix 3 [ ] STREAM CONNECTED 52313
unix 2 [ ] DGRAM 16844
unix 3 [ ] STREAM CONNECTED 15768
unix 3 [ ] STREAM CONNECTED 13773 /run/systemd/journal/stdout
unix 3 [ ] STREAM CONNECTED 53750 /run/user/1000/at-spi/bus
unix 3 [ ] STREAM CONNECTED 13868
unix 3 [ ] STREAM CONNECTED 15575 /run/dbus/system_bus_socket
unix 3 [ ] STREAM CONNECTED 12606 /run/user/1000/bus
unix 3 [ ] STREAM CONNECTED 12395 /run/systemd/journal/stdout
unix 3 [ ] STREAM CONNECTED 9041
unix 3 [ ] STREAM CONNECTED 52483
unix 3 [ ] STREAM CONNECTED 17299
unix 3 [ ] STREAM CONNECTED 47922
unix 3 [ ] STREAM CONNECTED 52231
unix 3 [ ] STREAM CONNECTED 14764 /run/dbus/system_bus_socket
unix 3 [ ] STREAM CONNECTED 13738
unix 3 [ ] STREAM CONNECTED 15429
unix 3 [ ] STREAM CONNECTED 51981
unix 3 [ ] STREAM CONNECTED 14040
unix 3 [ ] DGRAM CONNECTED 10073
unix 3 [ ] STREAM CONNECTED 17068

Practical No. 03

Setup a WAN which contains wired as well as wireless LAN by using a packet tracer tool. Demonstrate transfer of a packet from LAN 1 (wired LAN) to LAN2 (Wireless LAN).

```
r1#show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0/0   192.168.1.1     YES manual up                    up
GigabitEthernet0/0/1   192.168.1.1     YES manual administratively down down
GigabitEthernet0/0/2   unassigned      YES unset  administratively down down
Vlan1                  unassigned      YES unset  administratively down down
r1#config t
Enter configuration commands, one per line.  End with CNTL/Z.
r1(config)#interface GigabitEthernet0/0/1
r1(config-if)#ip address 192.168.2.1 255.255.255.0
r1(config-if)#no shutdown

r1(config-if)#
%LINK-5-CHANGED: Interface GigabitEthernet0/0/1, changed state to up

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0/1, changed state to up

r1(config-if)#exit
r1(config)#exit
r1#
%SYS-5-CONFIG_I: Configured from console by console
show ip interface brief
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0/0   192.168.1.1     YES manual up                    up
GigabitEthernet0/0/1   192.168.2.1     YES manual up                    up
GigabitEthernet0/0/2   unassigned      YES unset  administratively down down
Vlan1                  unassigned      YES unset  administratively down down
r1#
%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0/1, changed state to down

%LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0/1, changed state to up
```

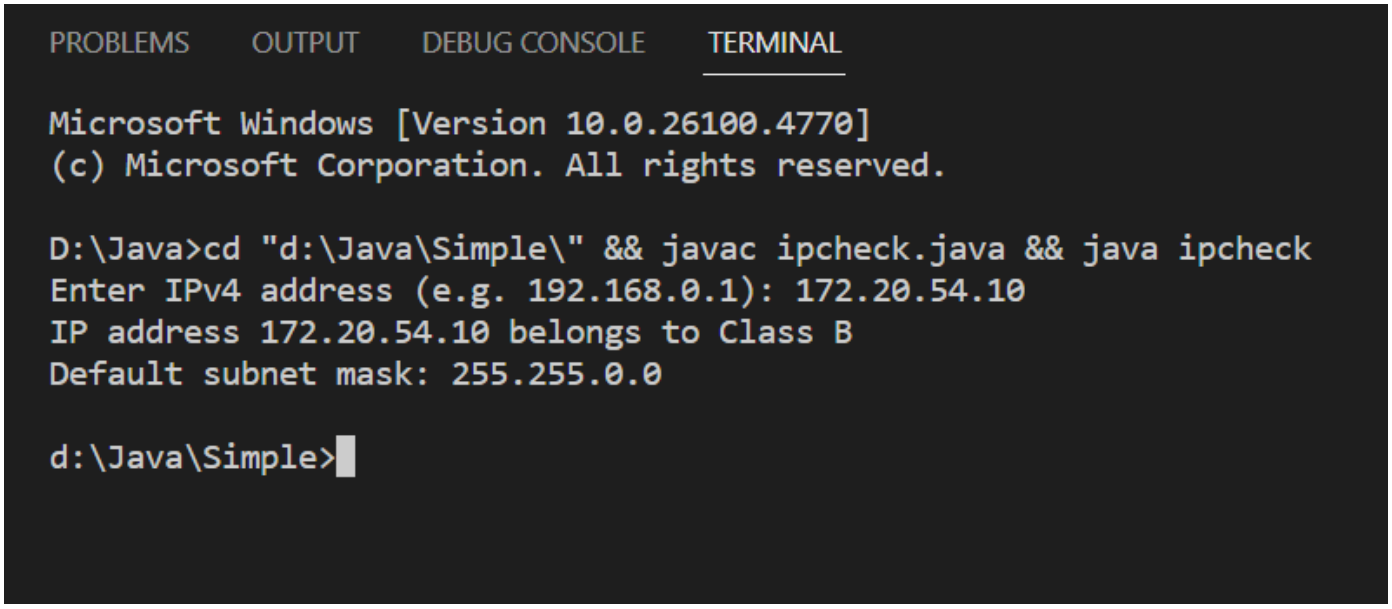


Practical No. 04

Write a program to demonstrate Sub-netting and find subnet masks.

```
import java.util.*;

public class ipcheck {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.print("Enter IPv4 address (e.g. 192.168.0.1): ");
      String ip = sc.nextLine().trim();

      if (!isValidIPv4(ip)) {
         System.out.println("Invalid IPv4 address format.");
         return;
      }

      int firstOctet = Integer.parseInt(ip.split("\\.")[0]);
      char ipClass = getClassOfIp(firstOctet);

      String subnetMask = getDefaultSubnetMask(ipClass);

      System.out.println("IP address " + ip + " belongs to Class " + ipClass);
      System.out.println("Default subnet mask: " + subnetMask);
   }

   static boolean isValidIPv4(String ip) {
      String[] parts = ip.split("\\.");
      if (parts.length != 4) return false;
      for (String part : parts) {
         if (part.length() == 0) return false;
         if (part.length() > 1 && part.charAt(0) == '0') return false;
         try {
            int num = Integer.parseInt(part);
            if (num < 0 || num > 255) return false;
         } catch (NumberFormatException e) {
            return false;
         }
      }
      return true;
   }

   static char getClassOfIp(int firstOctet) {
      if (firstOctet >= 0 && firstOctet <= 127) return 'A';
      else if (firstOctet >= 128 && firstOctet <= 191) return 'B';
      else if (firstOctet >= 192 && firstOctet <= 223) return 'C';
      else if (firstOctet >= 224 && firstOctet <= 239) return 'D';
      else return 'E';
   }
```

```
    static String getDefaultSubnetMask(char ipClass) {
        switch (ipClass) {
            case 'A': return "255.0.0.0";
            case 'B': return "255.255.0.0";
            case 'C': return "255.255.255.0";
            case 'D': return "Class D (multicast) has no default mask";
            case 'E': return "Class E (reserved) has no default mask";
            default: return "Unknown class";
        }
    }
}
```



Practical No. 05

Socket Programming using C/C++/Java.

a. TCP Client, TCP Server

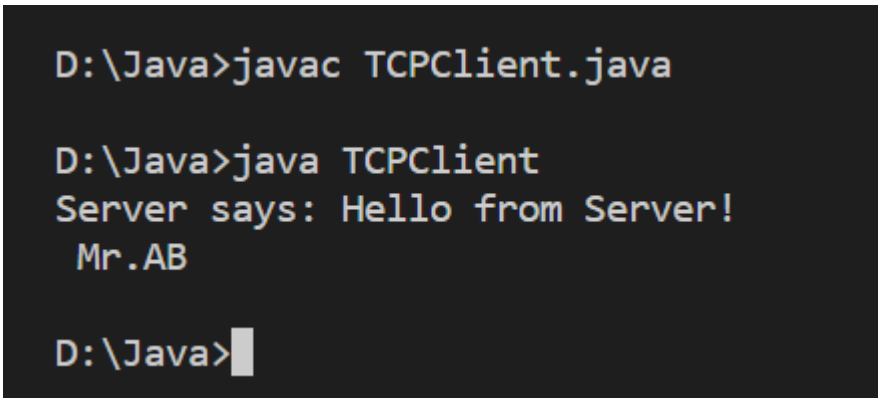
TCPServer:

```
import java.io.*;
import java.net.*;

public class TCPServer {
   public static void main(String[] args) throws IOException {
      ServerSocket serverSocket = new ServerSocket(5000); // Server at port 5000
      System.out.println("Server started. Waiting for client...");

      Socket socket = serverSocket.accept(); // Accept client connection
      System.out.println("Client connected.");

      BufferedReader input = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
      PrintWriter output = new PrintWriter(socket.getOutputStream(), true);

      String clientMessage = input.readLine();
      System.out.println("Client says: " + clientMessage);

      output.println("Hello from Server!");

      socket.close();
      serverSocket.close();
   }
}
```

TCPClient:

```
import java.io.*;
import java.net.*;

public class TCPClient {
   public static void main(String[] args) throws IOException {
      Socket socket = new Socket("localhost", 5000); // Connect to server

      BufferedReader input = new BufferedReader(new InputStreamReader(socket.getInputStream()));
      PrintWriter output = new PrintWriter(socket.getOutputStream(), true);

      output.println("Hello from Client!");

      String serverMessage = input.readLine();
      System.out.println("Server says: " + serverMessage);
      System.out.println(" Mr.AB ");

      socket.close();
   }
}
```

Output:-



b. UDP Client, UDP Serve

UDPClient:

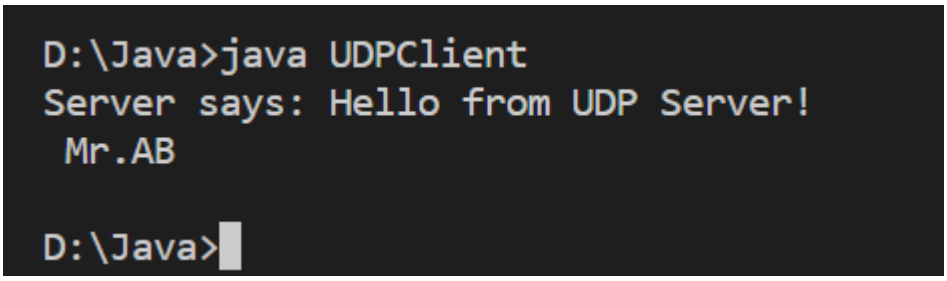
```
import java.net.*;

public class UDPClient {
   public static void main(String[] args) throws Exception {
      DatagramSocket socket = new DatagramSocket();

      InetAddress serverAddress = InetAddress.getByName("localhost");
      byte[] sendData = "Hello from UDP Client!".getBytes();

      DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, serverAddress,
6000);
      socket.send(sendPacket);

      byte[] receiveData = new byte[1024];
      DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
      socket.receive(receivePacket);

      String serverMsg = new String(receivePacket.getData(), 0, receivePacket.getLength());
      System.out.println("Server says: " + serverMsg);
      System.out.println(" Mr.AB " );

      socket.close();
   }
}
```

UDPServer:

```
import java.net.*;
```

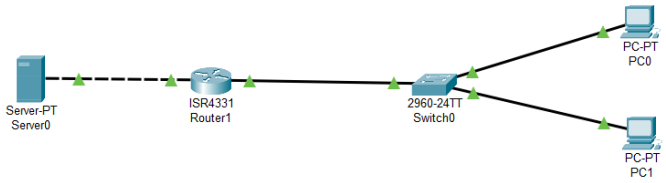
```
public class UDPServer {
   public static void main(String[] args) throws Exception {
      DatagramSocket socket = new DatagramSocket(6000);
      byte[] receiveData = new byte[1024];

      System.out.println("UDP Server started...");

      DatagramPacket receivePacket = new DatagramPacket(receiveData, receiveData.length);
      socket.receive(receivePacket);

      String clientMsg = new String(receivePacket.getData(), 0, receivePacket.getLength());
      System.out.println("Client says: " + clientMsg);

      String reply = "Hello from UDP Server!";
      byte[] sendData = reply.getBytes();

      InetAddress clientAddress = receivePacket.getAddress();
      int clientPort = receivePacket.getPort();

      DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length, clientAddress,
clientPort);
      socket.send(sendPacket);

      socket.close();
   }
}
```

Output:

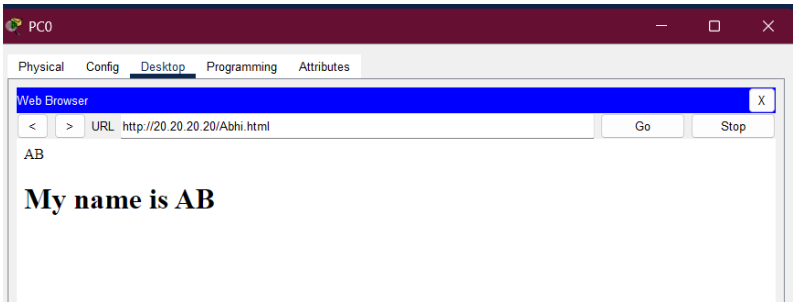


Practical No. 06

Study and Analyze the performance of HTTP, HTTPS and FTP protocol using Packet tracer tool.



Get the HTTP file

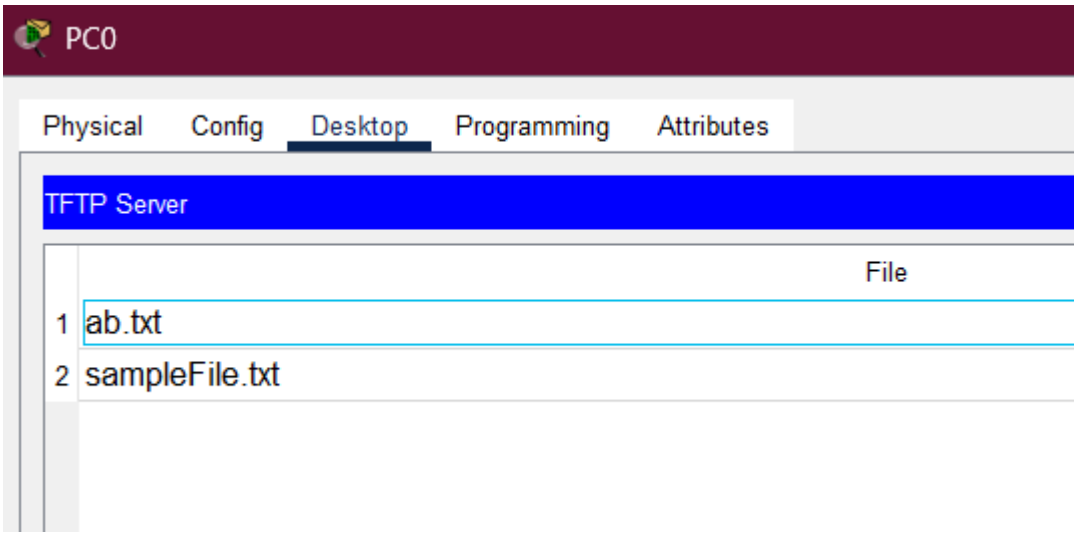


Put The txt file on server

```
C:\>ftp 20.20.20.20
Trying to connect...20.20.20.20
Connected to 20.20.20.20
220- Welcome to PT Ftp server
Username:a
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>put ab.txt

Writing file ab.txt to 20.20.20.20:
File transfer in progress...

[Transfer complete - 8 bytes]

8 bytes copied in 0.078 secs (102 bytes/sec)
ftp>
```

Get The txt file from server

```
C:\>ftp 20.20.20.20
Trying to connect...20.20.20.20
Connected to 20.20.20.20
220- Welcome to PT Ftp server
Username:a
331- Username ok, need password
Password:
230- Logged in
(passive mode On)
ftp>get ab.txt

Reading file ab.txt from 20.20.20.20:
File transfer in progress...

[Transfer complete - 8 bytes]

8 bytes copied in 0.001 secs (8000 bytes/sec)
ftp>
```



Practical No. 06

Name: Bhong Abhijit
Roll No. 29

Write a program using TCP socket for wired network for following
a. Say Hello to Each other

Server.java

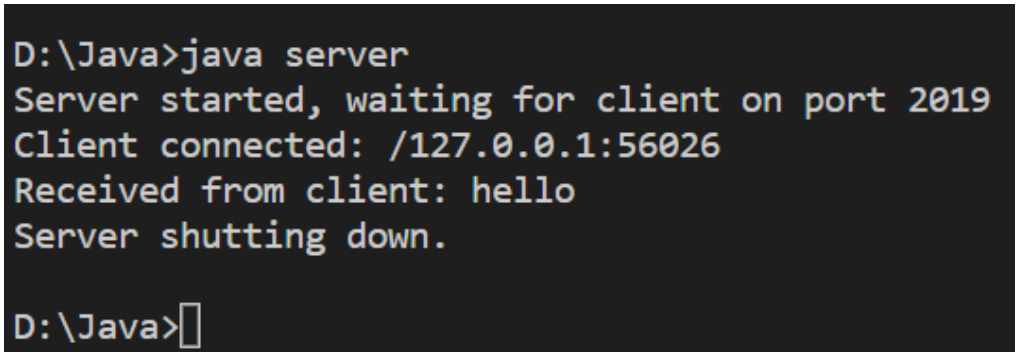
```
import java.net.ServerSocket;
import java.net.Socket;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.io.IOException;

public class server {
  public static void main(String[] args) {
    final int PORT = 2019;
    System.out.println("Server started, waiting for client on port " + PORT);
    try ( ServerSocket serverSocket = new ServerSocket(PORT);
         Socket clientSocket = serverSocket.accept();
         BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
         PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true) ) {

      System.out.println("Client connected: " + clientSocket.getRemoteSocketAddress());
      String userInput;
      while ((userInput = in.readLine()) != null) {
        System.out.println("Received from client: " + userInput);
        if (userInput.trim().equalsIgnoreCase("hello")) {
          out.println("hello client");
        } else {
          out.println("you did not say hello to server");
        }
      }
    } catch (IOException e) {
      System.err.println("IOException: " + e.getMessage());
      e.printStackTrace();
    }
    System.out.println("Server shutting down.");
  }
}
```

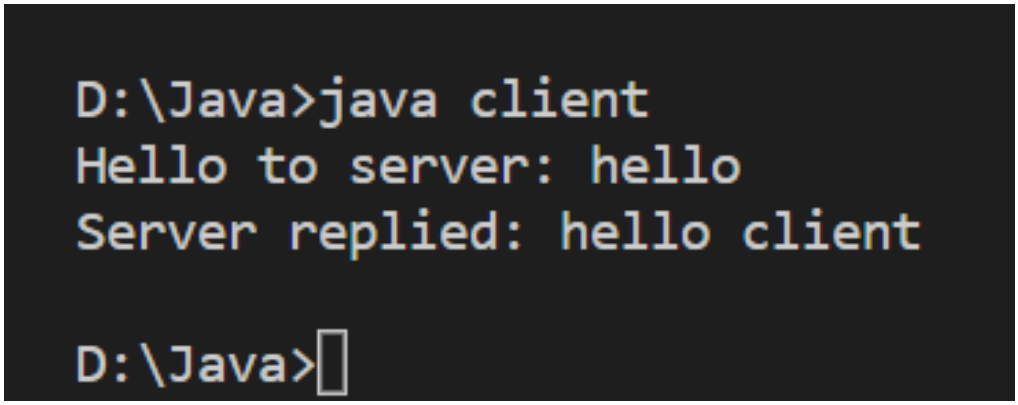
Client.java

```
import java.net.Socket;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
```

```
import java.io.IOException;

public class client {
  public static void main(String[] args) {
    final String HOST = "localhost";
    final int PORT = 2019;

    try ( Socket socket = new Socket(HOST, PORT);
          BufferedReader consoleReader = new BufferedReader(new InputStreamReader(System.in));
          BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
          PrintWriter out = new PrintWriter(socket.getOutputStream(), true) ) {

      System.out.print("Hello to server: ");
      String userInput = consoleReader.readLine();
      out.println(userInput);

      String response = in.readLine();
      System.out.println("Server replied: " + response);

    } catch (IOException e) {
      System.err.println("IOException: " + e.getMessage());
      e.printStackTrace();
    }
  }
}
```

Output:-





b. File transfer

FileClient:

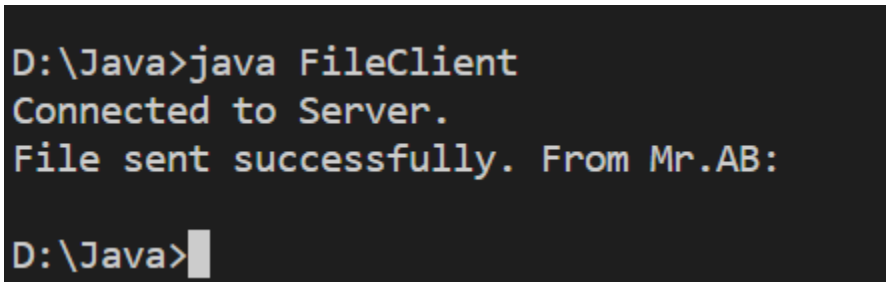
```
import java.io.*;
import java.net.*;

public class FileClient {
   public static void main(String[] args) throws IOException {
      Socket socket = new Socket("localhost", 6000);
      System.out.println("Connected to Server.");

      DataOutputStream dos = new DataOutputStream(socket.getOutputStream());
      String filePath = "test.txt"; // Example file in same folder

      // Send file name
      File file = new File(filePath);
      dos.writeUTF(file.getName());

      // Send file data
      FileInputStream fis = new FileInputStream(file);
      byte[] buffer = new byte[4096];
      int bytesRead;
      while ((bytesRead = fis.read(buffer)) != -1) {
         dos.write(buffer, 0, bytesRead);
      }

      System.out.println("File sent successfully. From Mr.AB:");


      fis.close();
      dos.close();
      socket.close();
   }
}
```

FileServer:

```
import java.io.*;
import java.net.*;

public class FileServer {
   public static void main(String[] args) throws IOException {
      ServerSocket serverSocket = new ServerSocket(6000);
      System.out.println("File Server started. Waiting for client...");

      Socket socket = serverSocket.accept();
      System.out.println("Client connected.");

      // Receive file name first
      DataInputStream dis = new DataInputStream(socket.getInputStream());
```

```
        String fileName = dis.readUTF();
        FileOutputStream fos = new FileOutputStream("Received_" + fileName);

        // Receive file data
        byte[] buffer = new byte[4096];
        int bytesRead;
        while ((bytesRead = dis.read(buffer)) != -1) {
            fos.write(buffer, 0, bytesRead);
        }

        System.out.println("File received successfully: Received_" + fileName);

        fos.close();
        dis.close();
        socket.close();
        serverSocket.close();
    }
}
```

Output:



Practical No. 07

Write a program using UDP Sockets to enable file transfer (Script, Text, Audio and Video one file each) between two machines.

Server.java

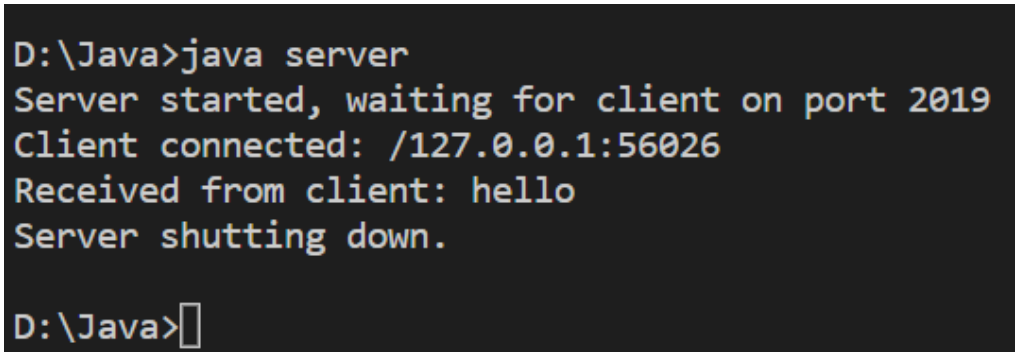
```
import java.net.ServerSocket;
import java.net.Socket;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.io.IOException;

public class server {
  public static void main(String[] args) {
    final int PORT = 2019;
    System.out.println("Server started, waiting for client on port " + PORT);
    try ( ServerSocket serverSocket = new ServerSocket(PORT);
          Socket clientSocket = serverSocket.accept();
          BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
          PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true) ) {

      System.out.println("Client connected: " + clientSocket.getRemoteSocketAddress());
      String userInput;
      while ((userInput = in.readLine()) != null) {
        System.out.println("Received from client: " + userInput);
        if (userInput.trim().equalsIgnoreCase("hello")) {
          out.println("hello client");
        } else {
          out.println("you did not say hello to server");
        }
      }
    } catch (IOException e) {
      System.err.println("IOException: " + e.getMessage());
      e.printStackTrace();
    }
    System.out.println("Server shutting down.");
  }
}
```

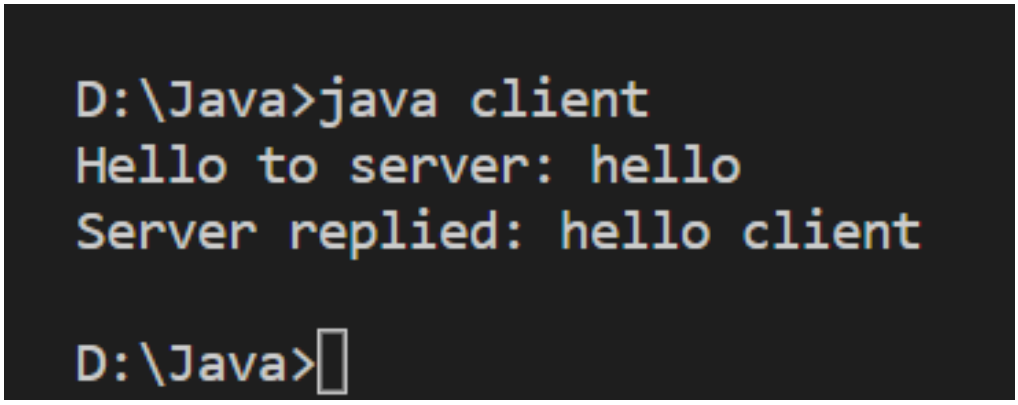
Client.java

```
import java.net.Socket;
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.io.IOException;

public class client {
```

```
 public static void main(String[] args) {
  final String HOST = "localhost";
  final int PORT = 2019;

  try ( Socket socket = new Socket(HOST, PORT);
       BufferedReader consoleReader = new BufferedReader(new InputStreamReader(System.in));
       BufferedReader in = new BufferedReader(new InputStreamReader(socket.getInputStream()));
       PrintWriter out = new PrintWriter(socket.getOutputStream(), true) ) {

    System.out.print("Hello to server: ");
    String userInput = consoleReader.readLine();
    out.println(userInput);

    String response = in.readLine();
    System.out.println("Server replied: " + response);

  } catch (IOException e) {
    System.err.println("IOException: " + e.getMessage());
    e.printStackTrace();
  }
 }
}
```

Output:-

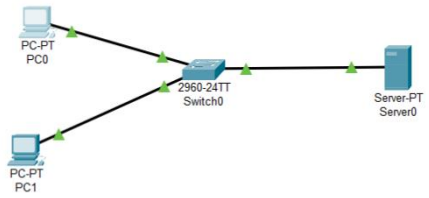


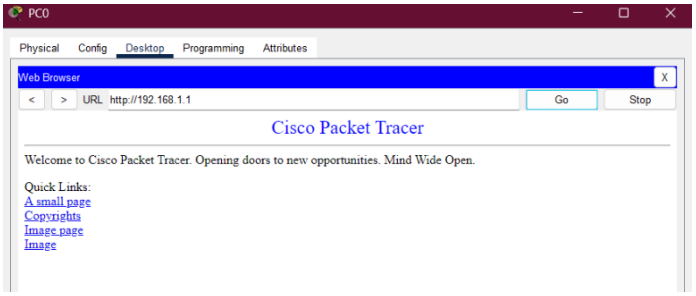


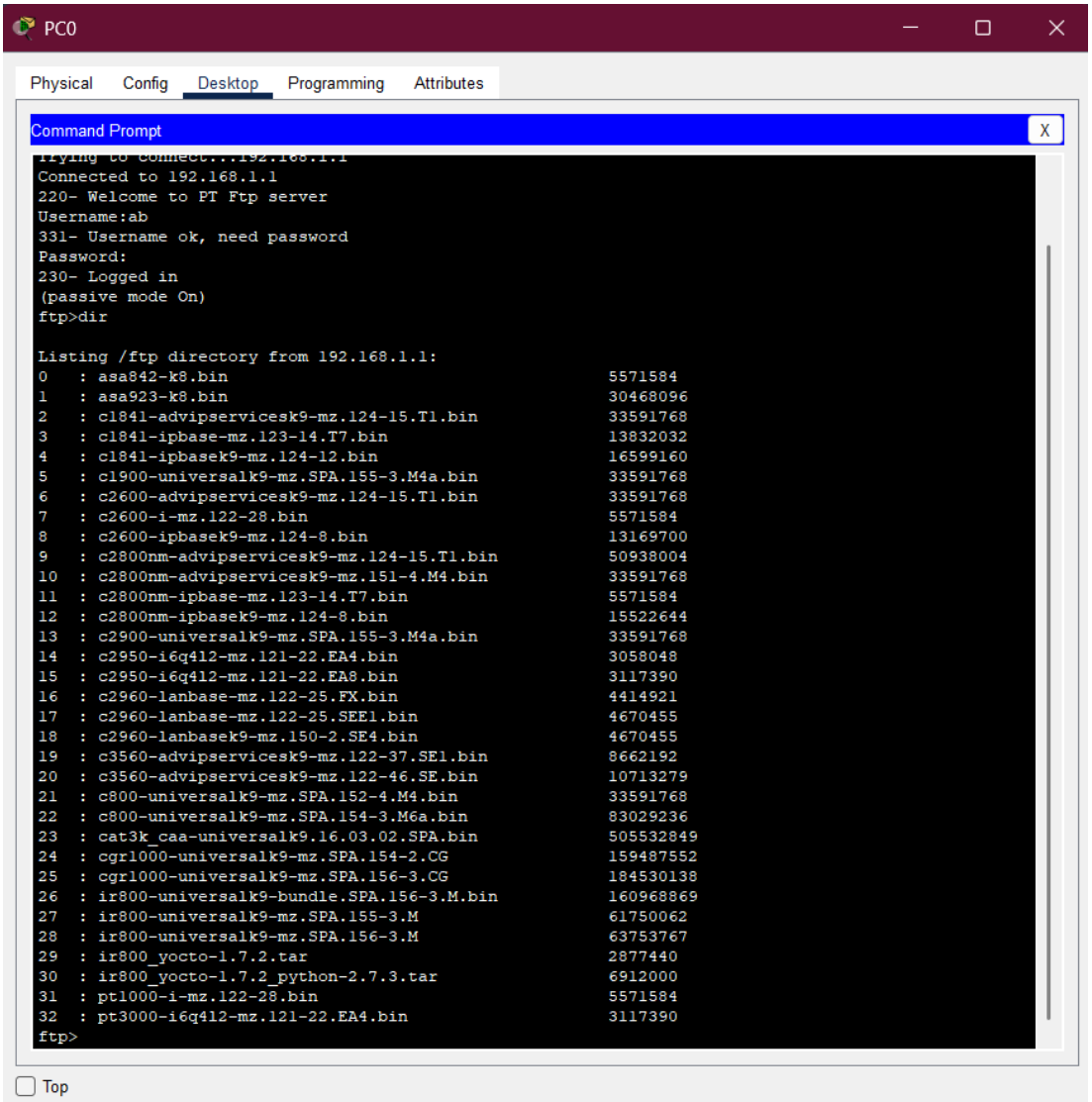
Practical No. 09

Study and Analyze the performance of HTTP, HTTPS and FTP protocol using Packet tracer tool.
Server.java

Output:-





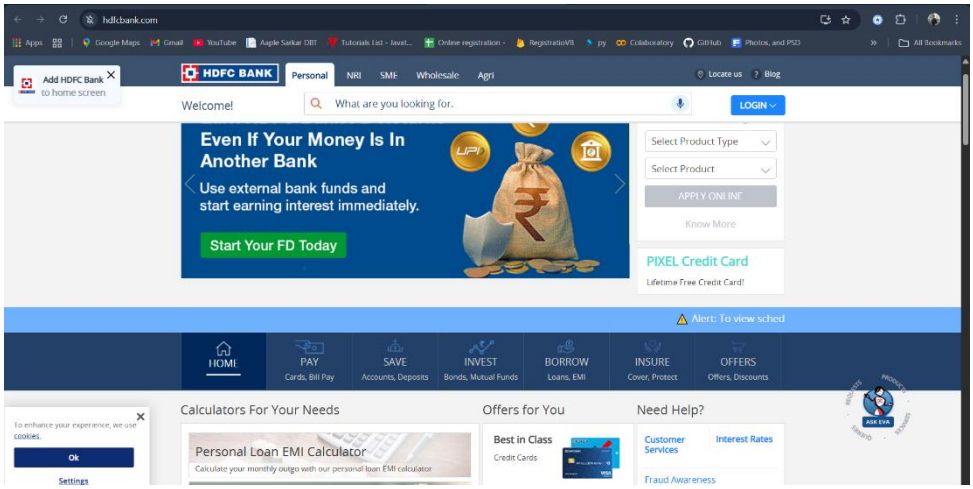


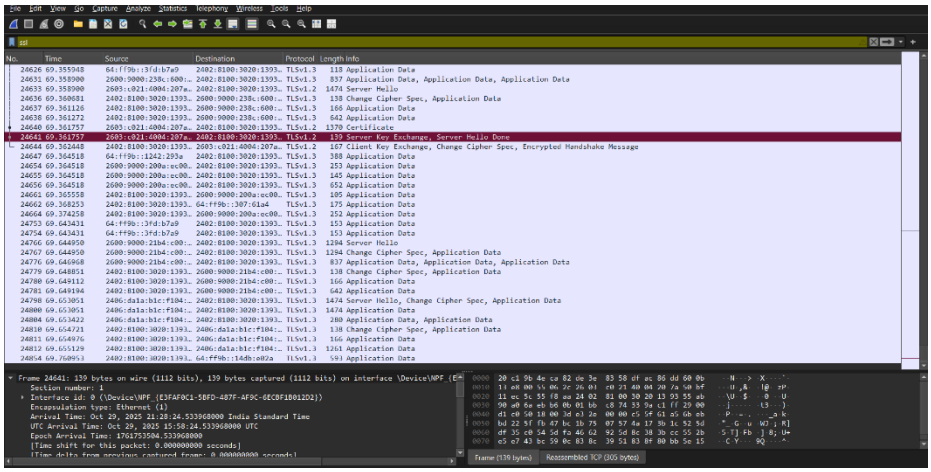
Practical No. 10

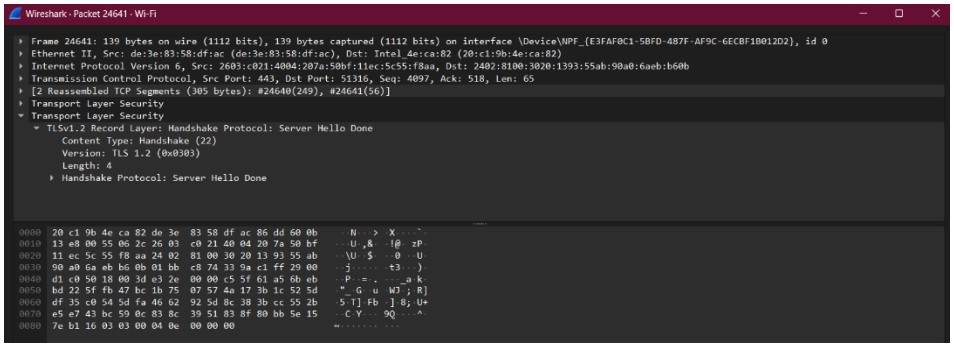
Name: Bhong Abhijit
Roll No. 29

To study the SSL protocol by capturing the packets using Wireshark tool while visiting any SSL secured website (banking, e-commerce etc.).

Output:-



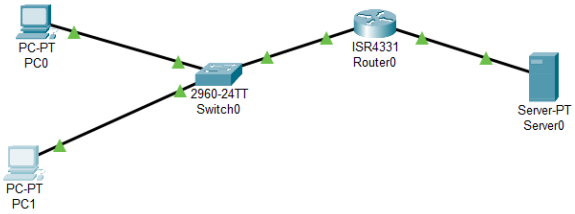


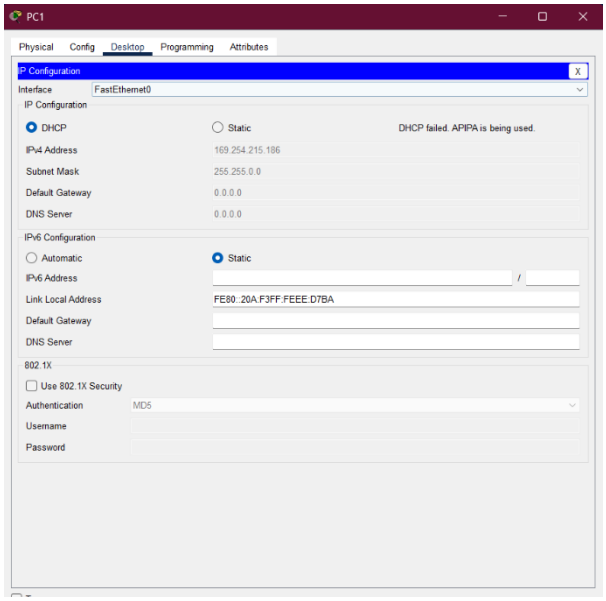


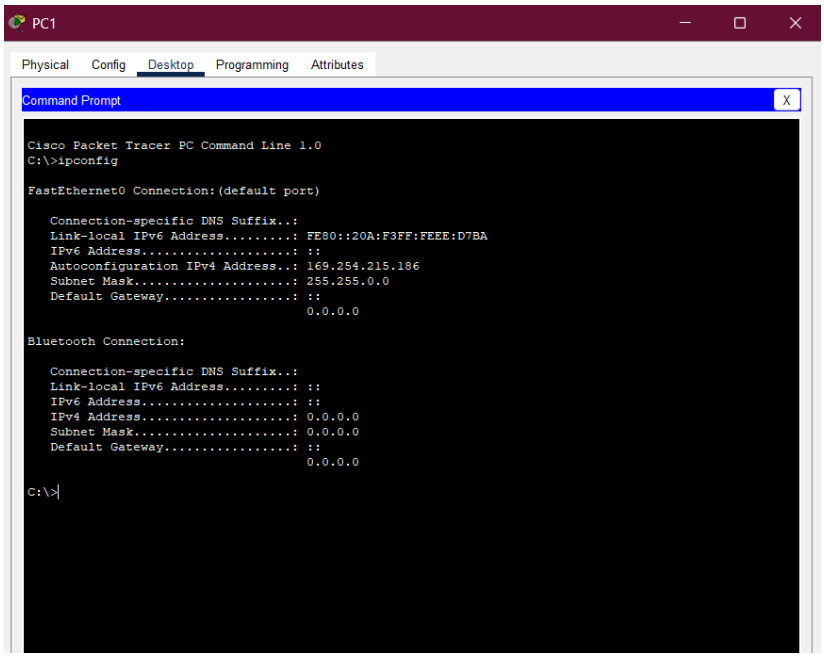
Practical No. 11

Installing and configuring DHCP server and assign IP addresses to client machines using DHCP server.

Output:-







Practical No. 12

Write a program for DNS lookup. Given an IP address input, it should return URL and vice versa.

Program:

```
import java.net.*;
import java.util.Scanner;

public class DNSLookup {
   public static void main(String[] args) {
      Scanner sc = new Scanner(System.in);
      System.out.println("Mr.AB");
      System.out.println("Enter '1' for Domain to IP lookup");
      System.out.println("Enter '2' for IP to Domain lookup");
      System.out.print("Your choice: ");
      int choice = sc.nextInt();
      sc.nextLine(); // consume newline

      try {
         if (choice == 1) {
            System.out.print("Enter domain name (e.g., www.google.com): ");
            String domain = sc.nextLine();
            InetAddress ip = InetAddress.getByName(domain);
            System.out.println("IP Address: " + ip.getHostAddress());
         }
         else if (choice == 2) {
            System.out.print("Enter IP address (e.g., 142.250.183.36): ");
            String ipStr = sc.nextLine();
            InetAddress addr = InetAddress.getByName(ipStr);
            System.out.println("Host Name: " + addr.getHostName());
         }
         else {
            System.out.println("Invalid choice!");
         }
      }
      catch (UnknownHostException e) {
         System.out.println("Unable to resolve host/IP: " + e.getMessage());
      }

      sc.close();
   }
}
```

Output:-

```
d:\Java>cd "d:\Java\" && javac DNSLookup.java && java DNSLookup
Mr.AB
Enter '1' for Domain to IP lookup
Enter '2' for IP to Domain lookup
Your choice: 1
Enter domain name (e.g., www.google.com): wwww.abphotovideographics.wordpress.com
IP Address: 64:ff9b:0:0:0:0:c000:4e0c

d:\Java>
```