| TITLE: NETWORK COMMANDS | GPREC-D/CS/EXPT-DCCN-01 |
|---|---|

**AIM: Study of basic network command and Network configuration commands.**

**1.ipconfig**

C:\Users\PGLAB-002>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

Connection-specific DNS Suffix . :
Link-local IPv6 Address ..... : fe80::bc0f:8c15:9c15:75c5%3
IPv4 Address..... : 192.168.6.31
Subnet Mask .....: 255.255.255.0
Default Gateway.....: 192.168.6.1

Tunnel adapter isatap.{866D805A-7C52-4A1A-9893-07EF6BF42621}:

Media State ..... Media disconnected
Connection-specific DNS Suffix . :

**2. ipconfig/all**

C:\Users\PGLAB-002>ipconfig/all

Windows IP Configuration

Host Name . . . . . : PGLAB-31
Primary Dns Suffix . . . . . :
Node Type ..... Hybrid
IP Routing Enabled..... No
WINS Proxy Enabled.....No

Ethernet adapter Ethernet:

Connection-specific DNS Suffix . :
Description .....Broadcom NetLink (TM) Gigabit Ethernet

Physical Address.....B8-AC-6F-37-0A-18
DHCP Enabled. .....No
Autoconfiguration Enabled.....Yes
Link-local IPv6 Address ..... : fe80::bc0f:8c15:9c15:75c5%3(Preferred)
IPv4 Address. . . . . : 192.168.6.31(Preferred)
Subnet Mask ..... : 255.255.255.0
Default Gateway..... : 192.168.6.1
DHCPv6 IAID ..... 62434415
DHCPv6 Client DUID. . . . . : 00-01-00-01-2A-41-BA-08-B8-AC-6F-37-0A-18

DNS Servers ..... : 192.168.6.1
NetBIOS over Tcpip. ..... Enabled

Tunnel adapter isatap.{866D805A-7C52-4A1A-9893-07EF6BF42621}:

Media State ..... Media disconnected
Connection-specific DNS Suffix . :
Description .....Microsoft ISATAP Adapter
Physical Address. . . . . : 00-00-00-00-00 00 00-E0
DHCP Enabled. .....No
Autoconfiguration Enabled.....Yes

**3. ping**

C:\Users\PGLAB-002>ping google.com

Pinging google.com [142.250.183.142] with 32 bytes of data:
Reply from 142.250.183.142: bytes=32 time=41ms TTL=59
Reply from 142.250.183.142: bytes=32 time=41ms TTL=59
Reply from 142.250.183.142: bytes=32 time=41ms TTL=59
Reply from 142.250.183.142: bytes=32 time=41ms TTL=59

Ping statistics for 142.250.183.142:
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 41ms, Maximum = 41ms, Average = 41ms

C:\Users\PGLAB-002>nslookup www.gmail.com
Server: UnKnown
Address: 192.168.6.1

Non-authoritative answer:
Name: www.gmail.com
Addresses: 2404:6800:4009:810::2005

142.250.199.133

**4. tracert**

C:\Users\PGLAB-002>tracert www.gmail.com

Tracing route to www.gmail.com [142.250.182.229]
over a maximum of 30 hops:

1 <1 ms <1 ms <1 ms 192.168.6.1
2 37 ms 37 ms 37 ms bom07s29-in-f5.1e100.net [142.250.182.229]

Trace complete.

**5. hostname**

C:\Users\PGLAB-002>hostname
PGLAB-31

**6. systeminfo**

C:\Users\PGLAB-002>systeminfo

Host Name: PGLAB-31
OS Name: Microsoft Windows 8.1 Pro
OS Version: 6.3.9600 N/A Build 9600
OS Manufacturer: Microsoft Corporation
OS Configuration: Standalone Workstation
OS Build Type: Multiprocessor Free
Registered Owner: PGLAB-002
Registered Organization:
Product ID: 00260-00000-00001-AA576
Original Install Date: 18-06-2022, 11:35:11
System Boot Time: 04-08-2022, 10:36:33
System Manufacturer: Dell Inc.
System Model: OptiPlex 380
System Type: x64-based PC
Processor(s): 1 Processor(s) Installed.
[01] : Intel64 Family 6 Model 23 Stepping 10 GenuineIn
tel ~2933 Mhz
BIOS Version: Dell Inc. A07, 13-06-2012
Windows Directory: C:\Windows
System Directory: C:\Windows\system32
Boot Device: \Device\HarddiskVolume1

```
System Locale:            4009
Input Locale:             00004009
Time Zone:                (UTC+05:30) Chennai, Kolkata, Mumbai, New Delhi
Total Physical Memory:    5,020 MB
Available Physical Memory: 2,880 MB
Virtual Memory: Max Size: 5,852 MB
Virtual Memory: Available: 3,658 MB
Virtual Memory: In Use:   2,194 MB
Page File Location(s):    C:\pagefile.sys
Domain:                   WORKGROUP
Logon Server:             \\PGLAB-31
Hotfix(s):                179 Hotfix(s) Installed.
                          [01]: KB2899189_Microsoft-Windows-CameraCodec-Package
Network Card(s):          1 NIC(s) Installed.
                          [01]: Broadcom NetLink (TM) Gigabit Ethernet
                               Connection Name: Ethernet
                               DHCP Enabled:    No
                               IP address(es)
                               [01]: 192.168.6.31
                               [02]: fe80::bc0f:8c15:9c15:75c5
Hyper-V Requirements:     VM Monitor Mode Extensions: Yes
                          Virtualization Enabled In Firmware: Yes
                          Second Level Address Translation: No
                          Data Execution Prevention Available: Yes
```

**7. pathping**

```
C:\Users\PGLAB-002>pathping gmail.com

Tracing route to gmail.com [142.250.195.37]
over a maximum of 30 hops:
  0 PGLAB-31 [192.168.6.31]
  1 192.168.6.1
  2 maa03s37-in-f5.1e100.net [142.250.195.37]

Computing statistics for 50 seconds...
            Source to Here This Node/Link
Hop  RTT    Lost/Sent = Pct Lost/Sent = Pct  Address
  0                                          PGLAB-31 [192.168.6.31]
                                   0/ 100 =  0%  |
  1    0ms     0/ 100 =  0%     0/ 100 = 0% 192.168.6.1
                                   1/ 100 =  1%  |
  2   16ms     1/ 100 =  1%     0/ 100 = 0% maa03s37-in-f5.1e100.net [142.250.
195.37]
```

Trace complete.

**8. netstat**

```
C:\Users\PGLAB-002>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    192.168.6.31:53026     ec2-35-163-171-85:https  ESTABLISHED
```

**9. arp**

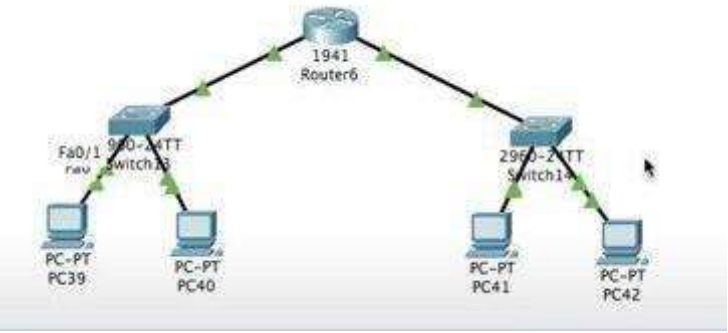
```
C:\Users\PGLAB-002>arp -a

Interface: 192.168.6.31 --- 0x3
  Internet Address      Physical Address      Type
  192.168.6.1           2c-c8-1b-69-97-12     dynamic
  192.168.6.255         ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
```

| TITLE: Network Model | GPREC-D/CS/EXPT-DCCN-02 |
|---|---|

**AIM: Create a network models using packet tracer.**

***TOPOLOGY**




**Step 1: Launch Packet Tracer.**

Launch Packet Tracer on your PC or laptop computer .Double click on the Packet Tracer icon on desktop or navigate to the directory that contains the Packet Tracer executable file and launch Packet Tracer. Packet Tracer should open with a blank default Logical topology workspace

**Step 2: Build the topology**

- Add network devices to the workspace

Using the device selection box, add the network devices to the workspace as shown in the topology diagram. To place a device onto the workspace, first choose a device type from the Device-Type Selection box. Then, click on the desired device model from the Device-Specific Selection box. Finally, click on a location in the workspace to put your device in that location. If you want to cancel your selection, click the Cancel icon for that device. Alternatively, you can click and drag a device from the Device-Specific Selection box onto the workspace.

- Change display names of the network devices.

To change the display names of the network devices click on the device icon on the Packet Tracer Logical workspace, then click on the Config tab in the device configuration window. Type the new name of the device into the Display Name box

- Add the physical cabling between devices on the workspace

Using the device selection box, add the physical cabling between devices on the workspace as shown in the topology diagram. The PC will need a copper straight-through cable to connect to the switch.

1. Select the copper straight-through cable in the device selection box and attach it to the FastEthernet 0 interface of the PC1 and the FastEthernet 0/1 interface of the switch0
2. Select the copper straight-through cable in the device selection box and attach it to the FastEthernet 0 interface of the PC2 and the FastEthernet 0/2 interface of the switch0
3. Select the copper straight-through cable in the device selection box and attach it to the FastEthernet 0 interface of the PC3 and the FastEthernet 0/1 interface of the switch1
4. Select the copper straight-through cable in the device selection box and attach it to the FastEthernet 0 interface of the PC4 and the FastEthernet 0/2 interface of the switch1
5. Select the copper straight-through cable in the device selection box and attach it to the FastEthernet 0/3 interface of switch0 and the gigabitethernet 0/0/0 interface of the router
6. Select the copper straight-through cable in the device selection box and attach it to the FastEthernet 0/3 interface of switch1 and the gigabitethernet 0/0/1 interface of the router

**Step 3:Configure the Network Devices**

Configure PC1.

Click on PC1 on the Packet Tracer Logical workspace and select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the ip address as 10.1.1.1

Configure PC2.

Click on PC2 on the Packet Tracer Logical workspace and select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the ip address as 10.1.1.2

Configure PC3.

Click on PC3 on the Packet Tracer Logical workspace and select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the ip address as 192.168.1.1

Configure PC4.

Click on PC4 on the Packet Tracer Logical workspace and select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the ip address as 192.168.1.2

Configure router.

Click on router on the Packet Tracer Logical workspace and select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, select gigabitethernet 0/0/0 and give the ip address as 10.1.1.3 and enable the port status.

Click on router on the Packet Tracer Logical workspace and select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, select gigabitethernet 0/0/1 and give the ip address as 192.168.1.3 and enable the port status.

Click on PC1 select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the default gateway as 10.1.1.3

Click on PC2 select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the default gateway as 10.1.1.3

Click on PC3 select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the default gateway as 192.168.1.3

Click on PC4 select the Desktop tab and then the IP Configuration icon. In the IP Configuration window, give the default gateway as 192.168.1.3

**Step 4:Verifying Connectivity**

Click on simulation.Select packets and assigned to PC0(Source) in LAN1 and PC3(Destination) in LAN2 and click on play button

1. packet is moved from PC0 to switch0
2. packet is moved from switch0 to router
3. packet is moved from router to switch1
4. packet is moved from switch1 to PC3

Acknowledgement:

5. packet is moved from PC3 to switch1
6. packet is moved from switch1 to router
7. packet is moved from router to switch0
8. packet is moved from switch0 to PC1

| TITLE: SWITCH CONFIGURATION | GPREC-D/CS/EXPT-DCCN-03 |
|---|---|

**AIM: Perform an Initial Switch Configuration using packet tracer.**

**Topology**


**Objectives**

**Part 1: Verify the Default Switch Configuration**

**Part 2: Configure a Basic Switch Configuration**

**Part 3: Configure a MOTD Banner**

**Part 4: Save Configuration Files to NVRAM**

**Part 1: Verify the Default Switch Configuration**

**Step 1: Enter privileged mode.**

We can access all switch commands from privileged mode. However, because many of the privileged commands configure operating parameters, privileged access should be password-protected to preventunauthorized use.

The privileged EXEC command set includes those commands contained in user EXEC mode, as well as the **configure** command through which access to the remaining command modes are gained.

a. Click **S1** and then the **CLI** tab. Press **<Enter>**.

b. Enter privileged EXEC mode by entering the **enable** command:

Switch> **enable**

Switch#

Notice that the prompt changed in the configuration to reflect privileged EXEC mode.

**Step 2: Examine the current switch configuration.**

Enter the **show running-config** command.

Switch# **show running-config**

**Output:**

```
Current configuration : 1080 bytes
!
version 12.2
no service timestamps log datetime msec
no service timestamps debug datetime msec
no service password-encryption
!
                hostname Switch
interface FastEthernet0/1
!
interface FastEthernet0/2
!
interface FastEthernet0/3
!
interface FastEthernet0/4
interface GigabitEthernet0/1
!
interface GigabitEthernet0/2
!
interface Vlan1
no ip address
shutdown
!
!
!
!
line con 0
!
line vty 0 4
login
line vty 5 15
login
!
!
!
!
end
```

**Part 2: Create a Basic Switch Configuration**

**Step 1: Assign a name to a switch.**

Switch#**configure terminal**
Switch(config)#**hostname S1**
S1(config)# **exit**
S1#

**Step 2: Secure access to the console line.**

To secure access to the console line, access config-line mode and set the console password to **letmein**.

*S1#* **configure terminal**
Enter configuration commands, one per line. End with CNTL/Z.
S1(config)# **line console 0**
S1(config-line)# **password letmein**
S1(config-line)#**login**
S1(config-line)#**exit**
S1(config)# **exit**
%SYS-5-CONFIG_I: Configured from console by consoleS1#

**Step 3: Verify that console access is secured.**

Exit privileged mode to verify that the console port password is in effect.

S1# **exit**
Switch con0 is now available
Press RETURN to get started.

User Access Verification
Password:
S1>

**Step 4: Secure privileged mode access.**

Set the **enable** password to **c1$c0**. This password protects access to privileged mode.

S1> **enable**
S1# **configure terminal**
S1(config)# **enable password c1$c0**
S1(config)# **exit**
%SYS-5-CONFIG_I: Configured from console by consoleS1#

**Step 5: Verify that privileged mode access is secure.**

a. Enter the **exit** command again to log out of the switch.

b. Press **<Enter>** and you will now be asked for a password:

User Access Verification

Password:

c. The first password is the console password configured for **line con 0**. Enter this password to return touser EXEC mode.

d. Enter the command to access privileged mode.

e. Enter the second password we configured to protect privileged EXEC mode.

f. Verify your configurations by examining the contents of the running-configuration file:

S1# **show running-config**

**Output:**

```
Current configuration : 1125 bytes
!
version 12.2
no service timestamps log datetime msec
no service timestamps debug datetime msec
no service password-encryption
!
hostname s1
!
enable password c1$c0
line con 0
password letmein
login
```

**Step 6: Configure an encrypted password to secure access to privileged mode.**

The **enable password** should be replaced with the newer encrypted secret password using the **enablesecret** command. Set the enable secret password to **itsasecret**.

S1# **config t**

S1(config)# **enable secret itsasecret**

S1(config)# **exit**

S1#

**Step 7: Verify that the enable secret password is added to the configuration file.**

Enter the **show running-config** command again to verify the new **enable secret** password is configured.

S1# **show run**

**Output:**

```
hostname s1
!
enable secret 5 $1$mERr$ILwq/b7kc.7X/ejA4Aosn0
enable password c1$c0
```

**Step 8: Encrypt the enable and console passwords.**

As noticed in Step 7, the **enable secret** password was encrypted, but the **enable** and **console**passwords were still in plain text. We will now encrypt these plain text passwords using the **servicepassword-encryption** command.

S1# **config t**

S1(config)# **service password-encryption**

S1(config)# **exit**

**Part 3: Configure a MOTD Banner**

**Step 1: Configure a message of the day (MOTD) banner.**

The Cisco IOS command set includes a feature that allows to configure messages that anyone logging onto the switch sees. These messages are called message of the day, or MOTD banners. Enclose the bannertext in quotations or use a delimiter different from any character appearing in the MOTD string.

S1# **config t**

S1(config)# **banner motd "This is a secure system. Authorized Access Only!"**
S1(config)# **exit**

%SYS-5-CONFIG_I: Configured from console by consoleS1#

## Part 4: Save Configuration Files to NVRAM

**Step 1: Verify that the configuration is accurate using the show run command.**

```
service password-encryption
!
hostname s1
!
enable secret 5 $1$mERr$ILwq/b7kc.7X/ejA4Aosn0
banner motd ^CThis is a secure system.Authorized Access only!^C
!
!
!
line con 0
password 7 082D495A041C0C19
login
```

**enable password 7 08221D0A0A49**

**Step 2: Save the configuration file.**

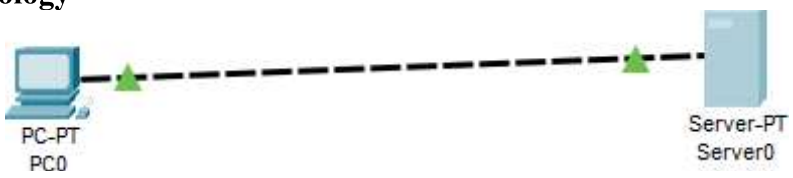
Completed the basic configuration of the switch. Now back up the running configuration file to NVRAM to ensure that the changes made are not lost if the system is rebooted or loses power.

S1#**copy running-config startup-config** Destination filename [startup-config]?**[Enter]**Building configuration...
[OK]

| TITLE: TCP-IP and OSI Models | GPREC-D/CS/EXPT-DCCN-04 |
|---|---|

**AIM:** Investigate the TCP-IP and OSI Models using packet tracer.

**Topology**




**Objectives**

Part 1: Examine HTTP Web Traffic

Part 2: Display Elements of the TCP/IP Protocol Suit

**Part 1: Examine HTTP Web Traffic**

**Step 1: Switch from Real time to Simulation mode**

In the lower right corner of the Packet Tracer interface are tabs to toggle between Real time and Simulation mode. PT always starts in Real time mode, in which networking protocols operate with realistic timings. However, a powerful feature of Packet Tracer allows the user to "stop time" by switching to Simulation mode. In Simulation mode, packets are displayed as animated envelopes, time is event driven, and the user can step through networking events.

a. Click the Simulation mode icon to switch from Real time mode to Simulation mode.

b. Select HTTP from the Event List Filters.

**Step 2: Generate web (HTTP) traffic.**

Currently the Simulation Panel is empty. There are six columns listed across the top of the Event List within the Simulation Panel. As traffic is generated and stepped through, events appear in the list. The Info column is used to inspect the contents of a particular event.

a. Click Web Client

b. Click the Desktop tab and click the Web Browser icon to open it.

c. In the URL field, enter server ip address and click Go. Because time in Simulation mode is event-driven, we must use the Capture/Forward button to display network events.

d. Click Capture/Forward four times. There should be four events in the Event List.

Event List

| Vis. | Time(sec) | Last Device | At Device | Type |
|---|---|---|---|---|
| | 0.002 | -- | PC0 | HTTP |
| | 0.003 | -- | PC0 | HTTP |
| | 0.004 | PC0 | Server0 | HTTP |
| | 0.005 | Server0 | PC0 | HTTP |

**Step 3: Explore the contents of the HTTP packet.**

a. Click the first colored square box under the Event List Info column.Ensure that the OSI Model tab is selected.

b. Under the Out Layers column, ensure that the Layer 7 box is highlighted.

OSI Model | Outbound PDU Details

At Device: PC0
Source: PC0
Destination: HTTP CLIENT

| In Layers | Out Layers |
|---|---|
| Layer7 | Layer 7: |
| Layer6 | Layer6 |
| Layer5 | Layer5 |
| Layer4 | Layer 4: TCP Src Port: 1033, Dst Port: 80 |
| Layer3 | Layer 3: IP Header Src. IP: 192.168.1.1, Dest. IP: 192.168.1.3 |
| Layer2 | Layer 2: Ethernet II Header 000A.418C.9E97 >> 00D0.D350.8024 |
| Layer1 | Layer 1: Port(s): |

1. The HTTP client sends a HTTP request to the server.

c. Click Next Layer. Layer 4 should be highlighted.

1. Sent segment information: the sequence number 1, the ACK number 1, and the data length 100.

d. Click Next Layer. Layer 3 should be highlighted.

1. The destination IP address is in the same subnet. The device sets the next-hop to destination.

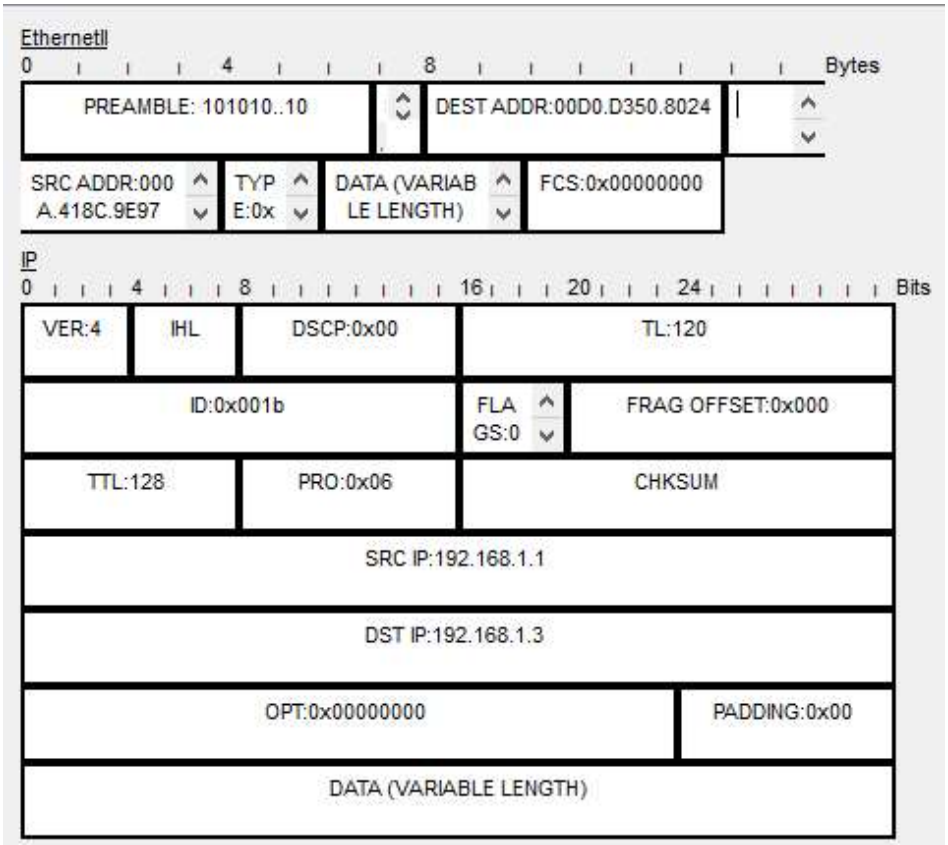
e. Click Next Layer.Layer 2 should be highlighted.

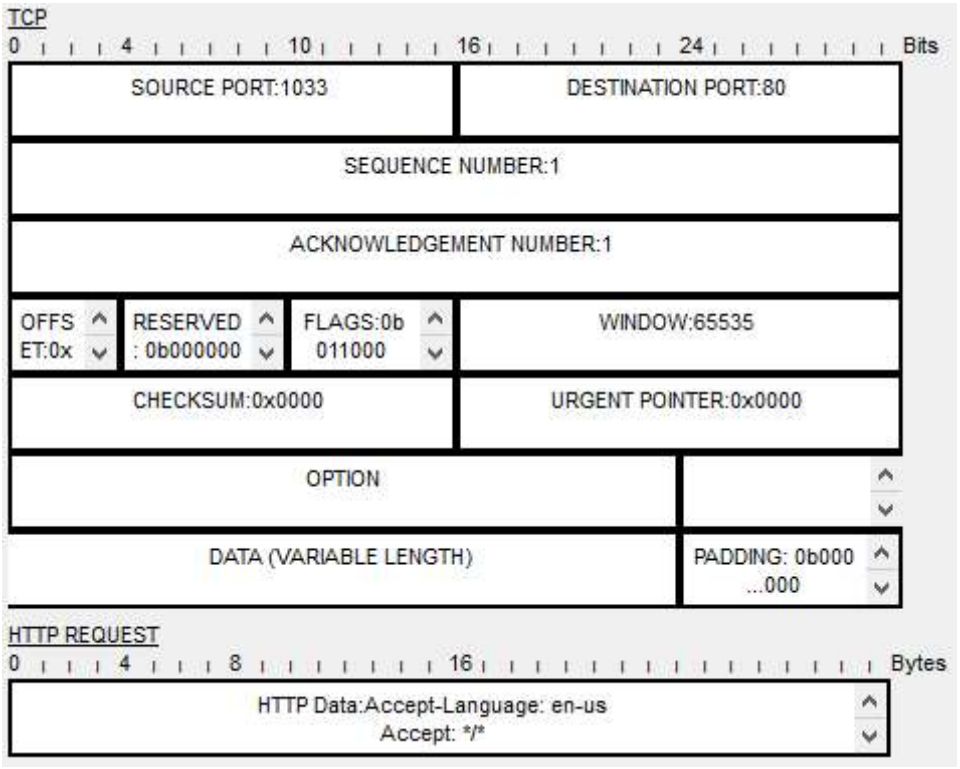
1. The next-hop IP address is a unicast. The ARP process looks it up in the ARP table.
2. The next-hop IP address is in the ARP table. The ARP process sets the frame's destination MAC address to the one found in the table.
3. The device encapsulates the PDU into an Ethernet frame.

f. Click Next Layer.Layer 1 should be highlighted.

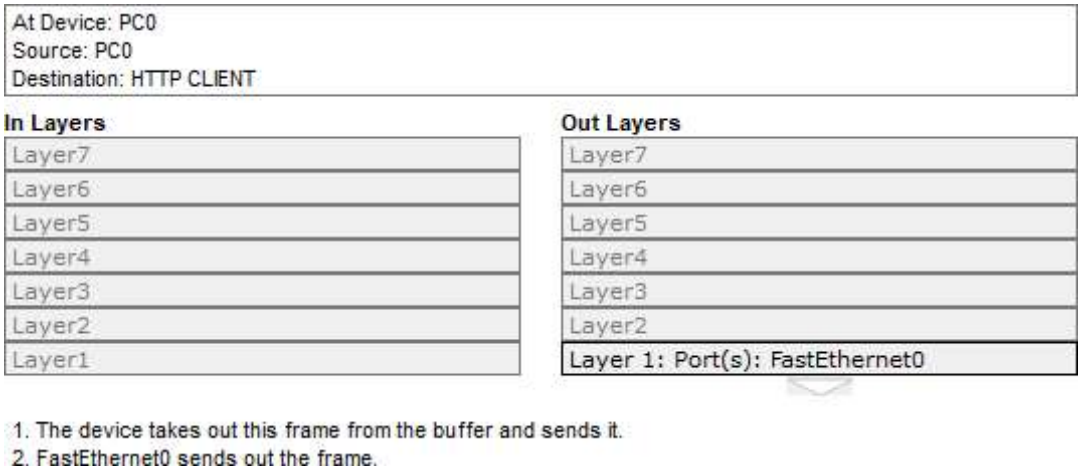
1. The port FastEthernet0 is sending another frame at this time. The device buffers the frame to be sent later.

g. Click the Outbound PDU Details tab.Information listed under the PDU Details is reflective of the layers within the TCP/IP model.

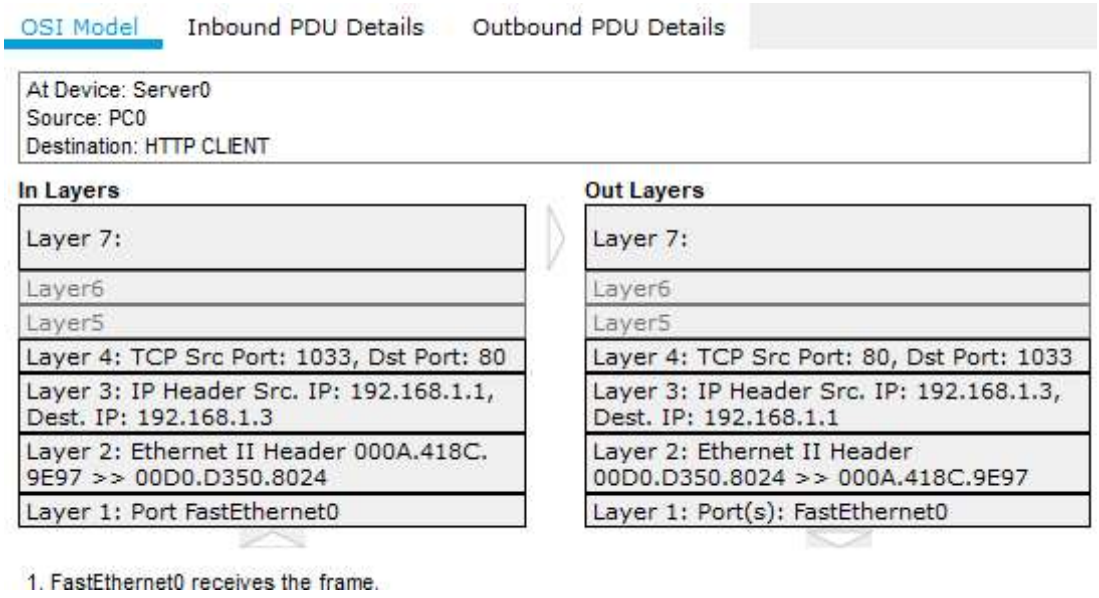





Click the next colored square box under the Event List > Info column. Only Layer 1 is active.The device is moving the frame from the buffer and placing it on to the network.




Advance to the next HTTP Info box within the Event List and click the colored square box. This window contains both In Layers and Out Layers.




Layer4

1. Sent segment information: the sequence number 1, the ACK number 101, and the data length 471.

Layer3

1. The destination IP address is in the same subnet. The device sets the next-hop to destination.

Layer 2

1. The next-hop IP address is a unicast. The ARP process looks it up in the ARP table.
2. The next-hop IP address is in the ARP table. The ARP process sets the frame's destination MAC address to the one found in the table.
3. The device encapsulates the PDU into an Ethernet frame.

Layer1

1. FastEthernet0 sends out the frame.

Receiver side

Layer1

1. FastEthernet0 receives the frame.

Layer2

1. The frame's destination MAC address matches the receiving port's MAC address, the broadcast address, or a multicast address.
2. The device decapsulates the PDU from the Ethernet frame.

Layer3

1. The packet's destination IP address matches the device's IP address or the broadcast address. The device de-encapsulates the packet.

Layer4

1. The device receives a TCP PUSH+ACK segment on the connection to 192.168.1.1 on port 1033.
2. Received segment information: the sequence number 1, the ACK number 1, and the data length 100.
3. The TCP segment has the expected peer sequence number.
4. TCP processes payload data.
5. TCP reassembles all data segments and passes to the upper layer.

Layer7

1. The server receives a HTTP request.

Click the next colored square box under the Event List > Info column.

OSI Model | Inbound PDU Details

At Device: PC0
Source: PC0
Destination: HTTP CLIENT

| In Layers | Out Layers |
|---|---|
| Layer 7: | Layer7 |
| Layer6 | Layer6 |
| Layer5 | Layer5 |
| Layer 4: TCP Src Port: 80, Dst Port: 1033 | Layer4 |
| Layer 3: IP Header Src. IP: 192.168.1.3, Dest. IP: 192.168.1.1 | Layer3 |
| Layer 2: Ethernet II Header 00D0.D350.8024 >> 000A.418C.9E97 | Layer2 |
| Layer 1: Port FastEthernet0 | Layer1 |

1. FastEthernet0 receives the frame.

Layer2

1. The frame's destination MAC address matches the receiving port's MAC address, the broadcast address, or a multicast address.
2. The device decapsulates the PDU from the Ethernet frame.

Layer3

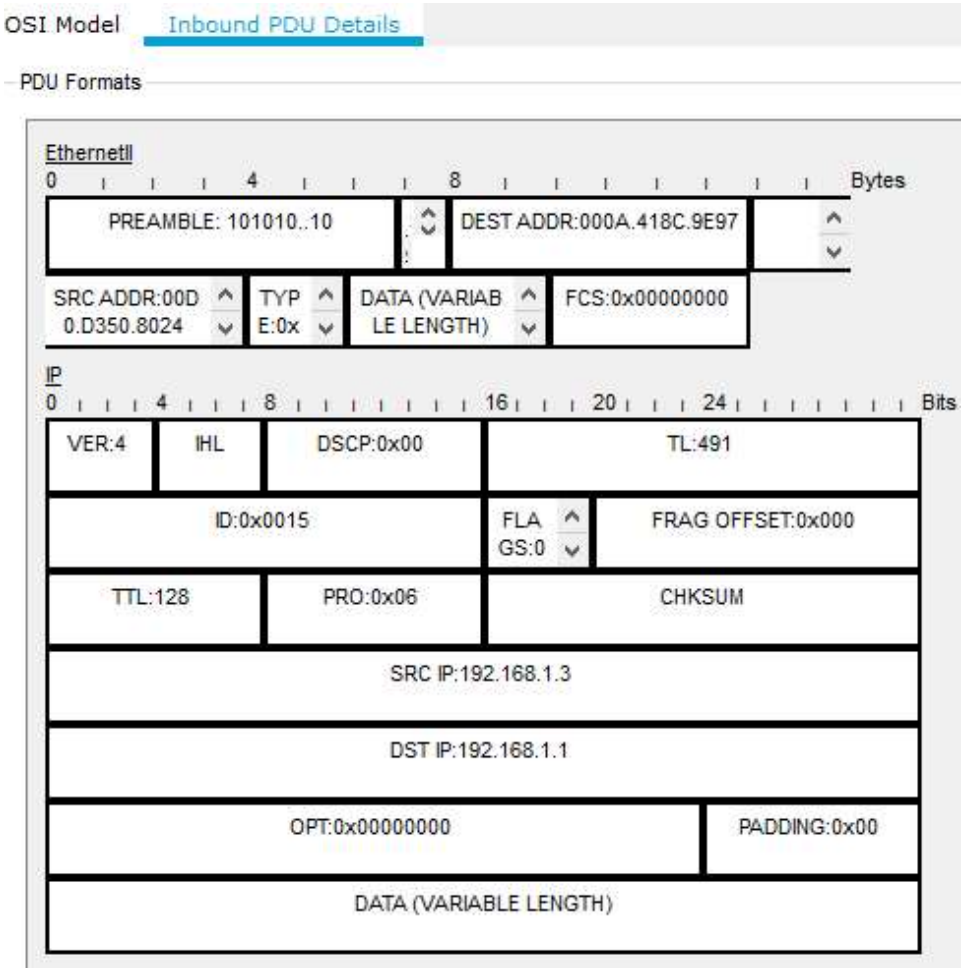
1. The packet's destination IP address matches the device's IP address or the broadcast address. The device de-encapsulates the packet.

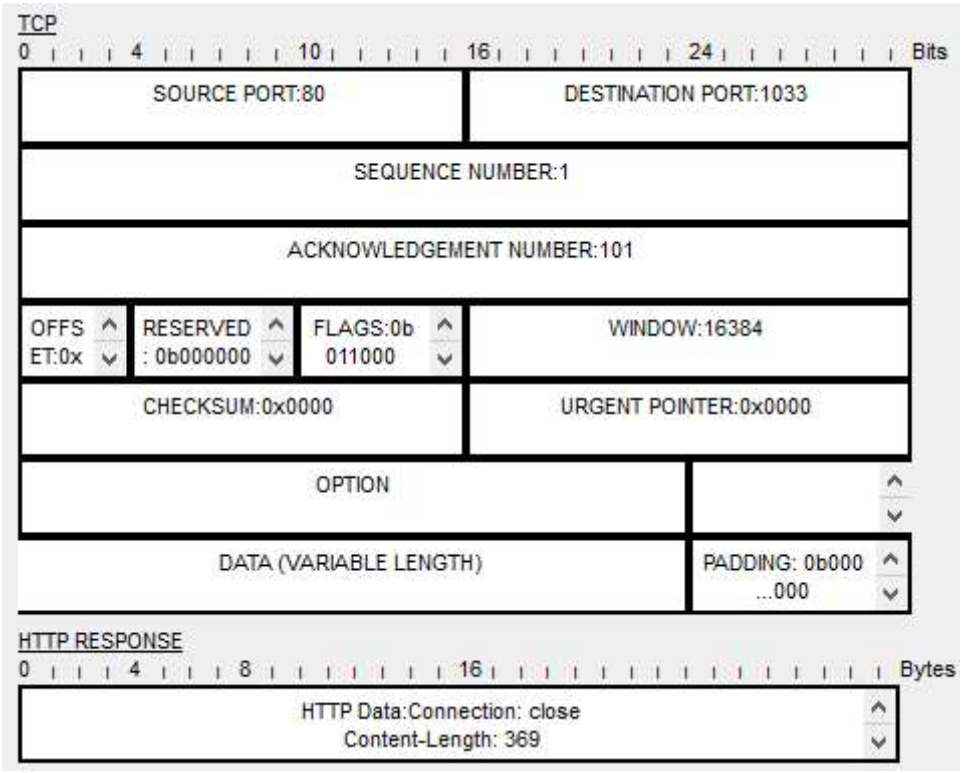
Layer4

1. The device receives a TCP PUSH+ACK segment on the connection to 192.168.1.3 on port 80.
2. Received segment information: the sequence number 1, the ACK number 101, and the data length 471.
3. The TCP segment has the expected peer sequence number.
4. The TCP segment has the expected ACK number. The device pops the last sent segment from the buffer.
5. TCP processes payload data.
6. TCP reassembles all data segments and passes to the upper layer.

Layer7

1. The HTTP client receives a HTTP reply from the server. It displays the page in the web browser.

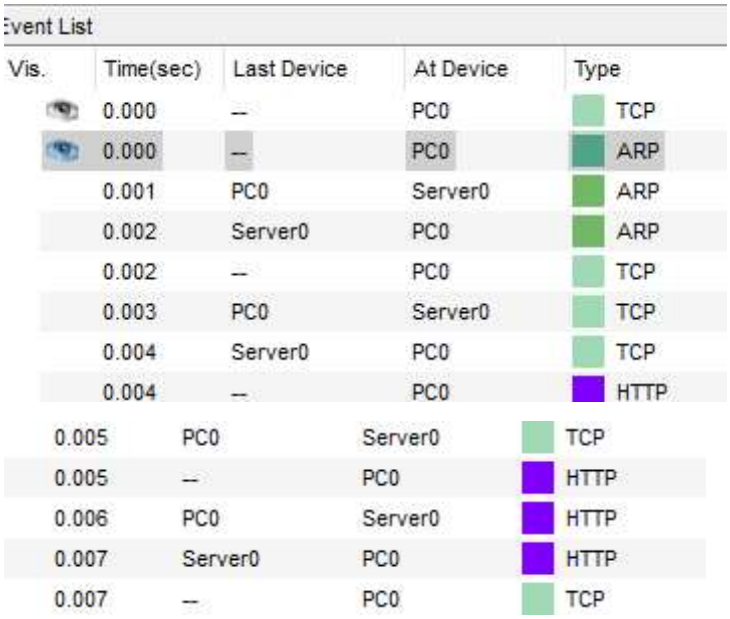

OSI Model
Inbound PDU Details
PDU Formats
EthernetII
0
4
8
Bytes
PREAMBLE: 101010..10
DEST ADDR:000A.418C.9E97
SRC ADDR:00D 0.D350.8024
TYP E:0x
DATA (VARIAB LE LENGTH)
FCS:0x00000000
IP
0
4
8
16
20
24
Bits
VER:4
IHL
DSCP:0x00
TL:491
ID:0x0015
FLA GS:0
FRAG OFFSET:0x000
TTL:128
PRO:0x06
CHKSUM
SRC IP:192.168.1.3
DST IP:192.168.1.1
OPT:0x00000000
PADDING:0x00
DATA (VARIABLE LENGTH)





**Part 2: Display Elements of the TCP/IP Protocol Suite**

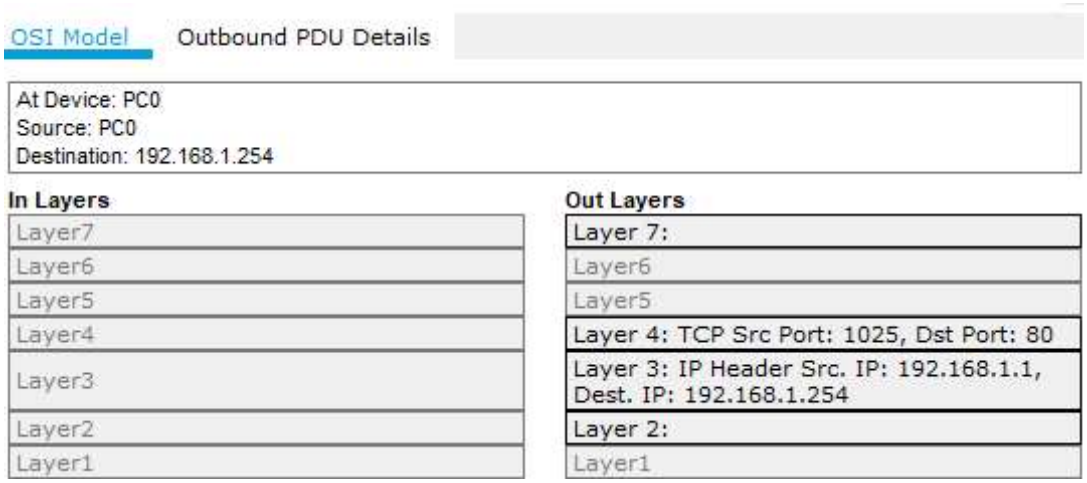
In the Event List Filters > Visible Events section, click Show All.



Event List

| Vis. | Time(sec) | Last Device | At Device | Type |
|---|---|---|---|---|
| | 0.000 | -- | PC0 | TCP |
| | 0.000 | -- | PC0 | ARP |
| | 0.001 | PC0 | Server0 | ARP |
| | 0.002 | Server0 | PC0 | ARP |
| | 0.002 | -- | PC0 | TCP |
| | 0.003 | PC0 | Server0 | TCP |
| | 0.004 | Server0 | PC0 | TCP |
| | 0.004 | -- | PC0 | HTTP |
| | 0.005 | PC0 | Server0 | TCP |
| | 0.005 | -- | PC0 | HTTP |
| | 0.006 | PC0 | Server0 | HTTP |
| | 0.007 | Server0 | PC0 | HTTP |
| | 0.007 | -- | PC0 | TCP |

Click the first colored square box TCP under the Event List Info column

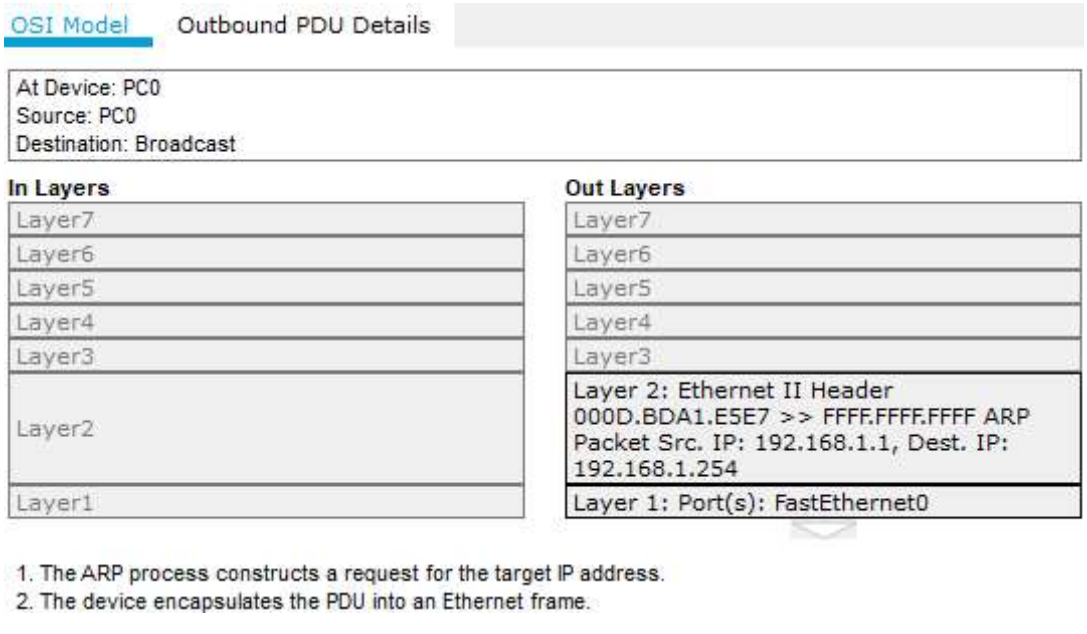


OSI Model | Outbound PDU Details

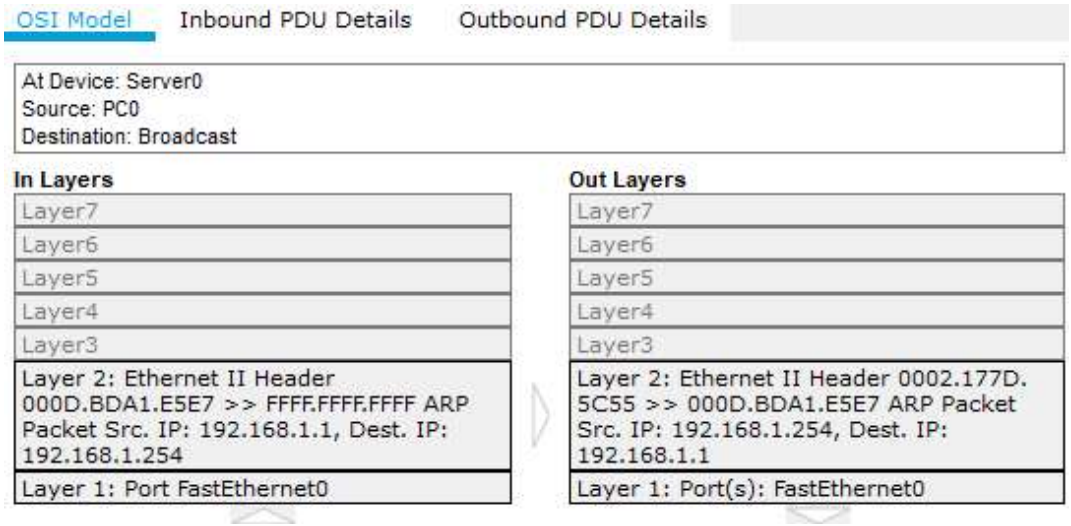
At Device: PC0
Source: PC0
Destination: 192.168.1.254

| In Layers | Out Layers |
|---|---|
| Layer7 | Layer 7: |
| Layer6 | Layer6 |
| Layer5 | Layer5 |
| Layer4 | Layer 4: TCP Src Port: 1025, Dst Port: 80 |
| Layer3 | Layer 3: IP Header Src. IP: 192.168.1.1, Dest. IP: 192.168.1.254 |
| Layer2 | Layer 2: |
| Layer1 | Layer1 |

Click the next colored square box ARP under the Event List Info column




Click the next colored square box ARP under the Event List Info column



CRC

```
import java.util.*;
public class CRC
{
    public static String concat(String s1,String s2 )
    {
        return (s1+s2);
    }
    public static String rem(String s1,String s2)
    {
        int n = s2.length();
        String temp1 = s1.substring(0,n-1);
        for(int i=n-1;i<s1.length();i++)
        {
            temp1+=s1.charAt(i);
            String temp2="";
            if(temp1.charAt(0)=='1')
            {
                temp2+= s2;
            }
            else
            {
                temp2+="0";
            }
            int num1 = Integer.parseInt(temp1, 2);
            int num2 = Integer.parseInt(temp2, 2);
            int result = num1 ^ num2;
            String resultBinary = Integer.toBinaryString(result);
            if(resultBinary.length()<n-1)
            {
                String add="";
                for(int j=0;j<n-resultBinary.length()-1;j++)
                {
                    add+="0";
                }
                resultBinary=add+resultBinary;
            }
            temp1=resultBinary;
        }
        return temp1;
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Dividend : ");
        String dividend = sc.next();
        System.out.println("Enter Divisor : ");
        String divisor = sc.next();
        String concatString="";
        for(int i=0;i<divisor.length()-1;i++)
        {
            concatString+="0";
        }
        String senderdividend=concat(dividend,concatString);
        System.out.println("Sender side : "+senderdividend);
        String temp1 = rem(senderdividend,divisor);
        System.out.println("Sender side remainder : "+temp1);
```

```
        String receiverdividend=concat(dividend,temp1);
        System.out.println("Receiver side : "+receiverdividend);
        String temp2 = rem(receiverdividend,divisor);
        System.out.println("Receiver side remainder : "+temp2);
        sc.close();
    }
}
```

OUTPUT

```
Enter Dividend :
1010011110
Enter Divisor :
10111
Sender side : 10100111100000
Sender side remainder : 1010
Receiver side : 10100111101010
Receiver side remainder : 0000
```

DIJKSTRA

```
import java.util.*;
public class Dijkstra
{
    public static int[] path(int graph[][],int start)
    {
        int n = graph.length;
        int distance[] = new int[n];
        boolean visited[] = new boolean[n];
        Arrays.fill(distance, 999);
        distance[start] =0;
        for(int i=0;i<n-1;i++)
        {
            int minIndex=-1;
            int minDistance = 999;
            for(int j=0;j<n;j++)
            {
                if(!visited[j] && distance[j]<minDistance) {
                minIndex=j;
                minDistance=distance[j];
                }
            }
            visited[minIndex]=true;
            for(int k=0;k<n;k++)
            {
                if(!visited[k] && graph[minIndex][k]!=999)
                {
                    int newDistance = distance[minIndex] + graph[minIndex][k];
                    if (newDistance < distance[k])
                    {
                        distance[k] = newDistance;
                    }
                }
            }
```

```
        }
        return distance;
    }
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Number of nodes : ");
        int nodes = sc.nextInt();
        System.out.println("Nodes: ");
        int graph[][] = new int[nodes][nodes];
        for(int i=0;i<nodes;i++)
        {
            for(int j=0;j<nodes;j++)
            {
                graph[i][j]=sc.nextInt();
            }
        }
        System.out.println("Enter Starting vertex : ");
        int start = sc.nextInt();
        int res[] = path(graph,start);
        for(int i=0;i<nodes;i++)
        {
            System.out.println("Minimum Distance from "+start+" node to"+i+" - "+res[i]);
        }
        sc.close();
    }
}
```

OUTPUT

```
Number of nodes :
6
Nodes:
0 2 1 999 999 999
2 0 7 999 8 4
1 7 0 7 999 3
999 999 7 0 8 4
999 8 999 8 0 5
999 4 3 4 5 0
Enter Starting vertex :
0
Minimum Distance from 0 node to 0 - 0
Minimum Distance from 0 node to 1 - 2
Minimum Distance from 0 node to 2 - 1
Minimum Distance from 0 node to 3 - 8
Minimum Distance from 0 node to 4 - 9
Minimum Distance from 0 node to 5 - 4
```

DVR

```
import java.util.*;
public class DVR
{
    public static void main(String[] args)
    {
```

```
        Scanner sc = new Scanner(System.in);
        System.out.println("Number of nodes : ");
        int n = sc.nextInt();
        int graph[][] = new int[n][n];
        System.out.println("Enter graph : ");
        for(int i=0;i<n;i++)
        {
            for(int j =0;j<n;j++)
            {
                graph[i][j]=sc.nextInt();
            }
        }
        for(int k=0;k<n;k++)
        {
            for(int i=0;i<n;i++)
            {
                for(int j=0;j<n;j++)
                {
                    graph[i][j]=Math.min(graph[i][j],graph[i][k]+graph[k][j]);
                }
            }
        }
        System.out.println("Final : ");
        for(int i=0;i<n;i++)
        {
            for(int j=0;j<n;j++)
            {
                System.out.print(graph[i][j]+" ");
            }
            System.out.println();
        }
        sc.close();
    }
}
```

OUTPUT

```
Number of nodes :
4
Enter graph :
0 2 999 1
2 0 3 7
999 3 0 11
1 7 11 0
Final :
0 2 5 1
2 0 3 3
5 3 0 6
1 3 6 0
```

Leaky Bucket

```
import java.util.*;
public class LB
{
    public static void main(String[] args)
    {
```

```
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter Bucket Size : ");
        int BucketSize = sc.nextInt();
        System.out.println("Enter Rate of dataflow : ");
        int ROD = sc.nextInt();
        System.out.println("Enter No. of packets : ");
        int n = sc.nextInt();
        System.out.println("Enter packets : ");
        int p[] = new int[n];
        for(int i=0;i<n;i++)
        {
            p[i]=sc.nextInt();
        }
        int sent=0,i=0,drop=0,packet,left=0;
        System.out.println("Packet"+"\tPacket sent"+"\tPacketsleft"+"\tPacket drop");
        do
        {
            packet=0;drop=0;
            if(i<n && sent<BucketSize && sent+p[i]<=BucketSize)
            {
                sent+=p[i];
                packet=p[i];
                i++;
            }
            else if(i<n && sent<BucketSize && sent+p[i]>BucketSize)
            {
                drop=p[i]-(BucketSize-sent);
                sent+=(BucketSize-sent);
                packet=p[i];
                i++;
            }
            if(sent>=ROD)
            {
                left=ROD;
                sent-=ROD;
            }
            else
            {
                left=sent;
                sent=0;
            }
            System.out.println(packet+"\t\t"+left+"\t\t"+sent+"\t\t"+drop);
        }while(sent!=0);
        sc.close();
    }
}
```

OUTPUT

```
Enter Bucket Size :
1000
Enter Rate of dataflow :
500
Enter No. of packets :
5
Enter packets :
600 900 800 1000 1900
Packet     Packet sent         Packet left         Packet drop
```

| | | | |
|---|---|---|---|
| 600 | 500 | 100 | 0 |
| 900 | 500 | 500 | 0 |
| 800 | 500 | 500 | 300 |
| 1000 | 500 | 500 | 500 |
| 1900 | 500 | 500 | 1400 |
| 0 | 500 | 0 | 0 |

DNS

```
import java.util.*;
import java.io.*;
import java.net.*;
public class DNS
{
   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      System.out.println("1.DNS\n2.Exit\nEnter your option : ");
      int op = sc.nextInt();
      while(true)
      {
         if(op==1)
         {
            System.out.println("Enter Host : ");
            String s = sc.next();
            try
            {
               InetAddress address = InetAddress.getByName(s);
               System.out.println("Host Name :"+address.getHostName());
               System.out.print("Host Address :"+address.getHostAddress());
            }
            catch(IOException ioe)
            {

            }
         }
         else
         {
            sc.close();
            System.exit(0);
         }
         System.out.println("\n1.DNS\n2.Exit\nEnter your option : ");
         op = sc.nextInt();
      }
   }
}
```

OUTPUT

```
1.DNS
2.Exit
Enter your option :
1
Enter Host :
www.gprec.ac.in
```

```
Host Name : www.gprec.ac.in
Host Address : 182.50.132.59
1.DNS
2.Exit
Enter your option :
1
Enter Host :
www.google.com
Host Name : www.google.com
Host Address : 142.250.192.100
1.DNS
2.Exit
Enter your option :
1
Enter Host :
www.amazon.com
Host Name : www.amazon.com
Host Address : 108.157.247.185
1.DNS
2.Exit
Enter your option :
2
```

CLIENT SERVER MODEL

CLIENT

```
import java.net.*;
import java.io.*;
class Client
{
  public static void main(String args[])throws Exception
  {
    Socket socket=new Socket("localhost",5000);
    System.out.println("Connected");
    DataInputStream in=new DataInputStream(socket.getInputStream());
    DataOutputStream out=new DataOutputStream(socket.getOutputStream());
    BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    String str="";
    while(!str.equals("Over"))
    {
      str=br.readLine();
      out.writeUTF(str);
      out.flush();
    }
    out.close();
    socket.close();
    }
}
```

OUTPUT

```
Connected
Hello
Hi
Gprec
Over
```

Server

```
import java.net.*;
import java.io.*;
class Server
{
    public static void main(String args[])throws Exception
    {
        ServerSocket server=new ServerSocket(5000);
        System.out.println("Server started");
        System.out.println("Waiting for a client ...");
        Socket socket=server.accept();
        System.out.println("Client accepted");
        DataInputStream in=new DataInputStream(socket.getInputStream());
        DataOutputStream out=new DataOutputStream(socket.getOutputStream());
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        String str="";
        while (!str.equals("Over"))
        {
            try
            {
                str= in.readUTF();
                System.out.println(str);
            }
            catch(IOException i)
            {
                System.out.println(i);
            }
        }
        try
        {
            System.out.println("Closing connection");
            socket.close();
            in.close();
        }
        catch(IOException i)
        {
            System.out.println(i);
        }
    }
}
```

OUTPUT

```
Server Started
Waiting for client .....
Client accecpted
HELLO
HI
GPREC
Over
Closing connection
```