**NETWORKING LAB  
CYCLE 1**

**I.** Familiarize the basic Linux commands that allow our system to communicate in Internet by performing the following experiments

1. View the configuration including addresses of your computer’s network interfaces

2. Test the network connectivity between your computer and several other computers

3. View the active TCP connections in the computer after visiting a website.

4. Find the hardware/MAC address of another computer in the network using ARP.

5. View the network configuration files

**II.** Familiarize the basic commands in Windows OS that allow our system to communicate in Internet by performing the following experiments

1. View the configuration including addresses of your computer’s network interfaces

2. Test the network connectivity between your computer and several other computers

3. View the active TCP connections in the computer after visiting a website.

4. Find the hardware/MAC address of another computer in the network using ARP.

5. View the network configuration files

**III.** Write the system calls used for creating sockets and transferring data between two nodes.

**IV.** Write a program to find the maximum, minimum and average of an array of integers using socket programming.

**V.** (a) Create three programs, two of which are clients to a single server. Client1 will send a string to the server process using stream socket. The server will reverse the string and send the result to Client2. Client2 prints the reversed string it receives and then all the processes terminate.

(b) Follow the same procedure as in part a except that the data type of the message should be integer and the server should square the integer before transmitting it to Client2.

(c) Write a socket program to enable Client1 to send a float value to the server. The server process should increase the value of the number it receives by a power of 1.5. The server should print both the value it receives and the value that it sends. Client2 should print the value it receives from the server.

**VI. Multi user chat server** using TCP

Write the client and server programs, where the server can exchange text with many client processes. A client process starts the communication with an input "start". After this, the client process waits for the answer from the server. If server permits, it can further send any text message (with restriction of not more than 25 words in a day). The communication goes on in this way until the client process sends the message " stop " to the server.

**VII.** Implement a Concurrent Time Server application using UDP to execute the program at a remote server. Client sends a time request to the server, server sends its system time back to the client. Client displays the result.

**VIII.** Implement a simple web proxy server that accepts HTTP requests and forwarding to remote servers and returning data to the client using TCP.

**CYCLE 2**

I. a) Implement Stop-and-Wait ARQ flow control protocol.

b) Implement Go-Back--N ARQ flow control protocol.

c) Implement Selective Repeat ARQ flow control protocol.

II. Implement the Distance Vector Routing algorithm.

III. Implement Link State Routing algorithm.

IV. Implement Simple Mail Transfer Protocol.

V. Implement File Transfer Protocol.

VI. Develop a concurrent file server which will provide the file requested by a client if it exists. If not, the server sends appropriate message to the client. Server should also send its process ID (PID) to clients for display along with the file or the message.

VII. Implement leaky bucket algorithm for congestion control.

**CYCLE 3**

I. Familiarization of Wireshark tool and packet capturing using it.

II. NS2 simulator and implementation of various protocols using NS2.