# NETWORK LAB

# LAB ASSIGNMENT for Week # 6

## Socket Programming - TCP Sockets

**Program # 1**

**String Reversal Server using TCP Sockets**

The problem is to implement a client/server user-level application using sockets API in C. The Server accepts strings from client (even multiple strings from client) and replies with reverse strings. For example, when client sends “MNNITALLD”, Server replies with “DLLATINNM”. Both server and client(s) have to output both sending & receiving strings on the terminal. The server and client processes should be made to run on different machines. Use TCP sockets.

**Program # 2**

The goal of this assignment is to implement a TCP client and server. Your TCP client/server will communicate over the network and exchange data. The user interface (i.e., what’s displayed to the user) should look the same for both the TCP applications.

Write a program named ***CalcClientTCP.c*** that performs the following functions:

1. Take a server hostname and a port number as command-line arguments.
   **Note:** Your client must resolve the hostname into an IP address.
2. Connect to the server at the given hostname and port using TCP.
3. Print the IP address and port of the server.
4. Ask the user for a simple arithmetic expression to calculate.
5. Send the expression to the server.
6. Read the answer from the server.
7. Display the answer to the user.
8. Repeat the steps 4-7 until the user enters the sentinel value given in the user prompt. When the user enters the sentinel, the client must close the connection to the server and quit.

Write a program named ***CalcServerTCP.c*** that performs the following functions:

1. Take a port number as a command-line argument.
2. Listen for a TCP connection on the port specified.
3. Print the IP address and port of the connected client.
4. Receive data from the client.

5. Evaluate the arithmetic expression.
6. Send the result back to the client.
7. If the connection is still open, repeat the steps 4-6 until the user presses Ctrl-C. If the connection is closed, repeat steps 2-6 until the user presses Ctrl-C.

**Example Output for Program # 2:**

The following is an example of execution of the TCP version assuming that the client is compiled with -

```
gcc CalcClientTCP.c –o CalcClientTCP
```

and the server is compiled with

```
gcc CalcServerTCP.c –o CalcServerTCP
```

```
Server                                  Client
% ./CalcServerTCP 50000                 % ./CalcClientTCP cash 50000
Server listening on port 50000          TCP client connected to 128.82.4.7 on port 50000
Client 128.82.4.75 on port 5983 connected   Enter an expression in the following format:
                                        operand1 operator operand2
                                        Valid operators are + - * / ^. To quit, enter -1.
                                        3.5 * -4
Received from client: 3.5 * -4
Sending to client: -14                  ANS: 3.5 * -4 = -14

                                        Enter an expression in the following format:
                                        operand1 operator operand2
                                        Valid operators are + - * / ^. To quit, enter -1.
                                        -1
Client closed connection                Bye!
Server listening on port 50000

                                        %
```