Program 2: Network Calculator

Assigned: Friday, September 15, 2006
Due: Monday, September 25, 2006 *before the start of class*

Objective

There are a number of objectives to this assignment. The first is to make sure you have some experience actually doing a simple socket programming assignment. Second, it will help you tune up your programming skills and prepare you for the other assignments in this course. Third, because you can use the Internet to look for examples, this assignment will help you see just how many network programming aids are available via the web. And finally, because this assignment has **many** small details, it will give you an awareness of just how much you have to pay attention to when we get to application-layer protocols. *This assignment may be completed using either C or C++.*

As with all programming assignments, your work should be your own, and your program should be documented well enough that the grader can understand how you designed your program. *No sharing of code is allowed.*

Assignment

The goal of this assignment is to implement a TCP client and server and a UDP client and server (for a total of ***four*** different programs). Your TCP or UDP 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 and UDP applications.

Write a program named `CalcClientTCP.c` (or `.cpp`) 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.
   Note: The user prompt must contain instructions for the user, including the proper format of the arithmetic expression and the sentinel value that will be used to indicate that the user wishes to quit.
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` (or `.cpp`) 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.
   Note: You must allow the user to add, subtract, divide, multiply, and raise a number to a power. You must also be able to handle negative numbers and numbers with decimals.
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.

Though not required, it may be helpful for you to output debugging information from your server.

The UDP application should perform all of the same functions as the TCP application, but use UDP as the transport protocol instead of TCP. The UDP client should be named `CalcClientUDP.c` (or `.cpp`), and the UDP server should be named `CalcServerUDP.c` (or `.cpp`).

After you have completed your programs, write the answers to the following questions in a file named `PROG2.txt`:

1. Start your TCP client application without the TCP server running. What happens? Why?
2. Start your UDP client application without the UDP server running. What happens? Why?

`PROG2.txt` should also contain your name, the course number, and the date

Grading Guidelines

You may use pieces of code from the Internet (including the course web pages) to help you do this assignment (*e.g.* basic socket code). However, this is just like citing a passage from a book, so if you copy code, you must cite it. To do this, put a comment at the beginning of your code that explains exactly what you have copied, who originally wrote it, and where it came from.

Make sure you do sufficient error handling such that a user cannot crash your client or server. For instance, what will you do if the user provides invalid input or does not provide any command-line arguments?

Below is a breakdown of points for this assignment:

- 20 pts: TCP client
- 20 pts: TCP server
- 20 pts: UDP client
- 20 pts: UDP server
- 20 pts: Documentation/Proper References, `PROG2.txt`

Example

The following is an example of execution of the TCP version assuming that the client is compiled with
```
g++ CalcClientTCP.c –o CalcClientTCP –lnsl –lsocket
```
and the server is compiled with
```
g++ CalcServerTCP.c –o CalcServerTCP –lnsl –lsocket
```

```
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
                                               %
```

<u>Submission</u>

Submit your **four (4) source code files and `PROG2.txt`** via Blackboard. See the course webpage (look under “Useful Links”) for detailed instructions on submitting assignments via Blackboard. You will lose points if you do not name your files as specified above.