Laboratory 5

This laboratory will result in your ability to build a simple program that passes data structures in messages between a client and server program.

Objective:

Write a client that sends a message containing two integers and a character representing the operation to a server. The server computes the numeric value for the result of the numbers and the given operation, and then returns a message containing a structure as the result. The client will pass the two numbers as integer members of an "input parameters" struct and the operation as a character value in the struct. The server will reply with a different struct containing a double, a flag representing whether or not an error occurred (or would have occurred if the operation had been attempted), and a character string of up to 128 bytes containing the error description. (You can use your imagination for the string content but try to cover all types of errors.)

The operation should be one of: '+', '-', 'x', and '/'. (Note: that is an 'x' not a '*'). The input integer values could be any valid integer. The server must check for division by 0. The client must validate the number of command-line arguments. If the number of command-line arguments does not equal 5 (exactly) then:

Display a usage message to stdout

exit with EXIT_FAILURE

Start the server in the background (use the '&' symbol) and print its process_id (PID). Then multiple client applications could be run in the foreground that use a command-line argument on the client application to pass the PID of the server which they want to use to perform the operation.

The client program will block until receiving the result message from the server program and then the client program will output the result (or error condition) to the console. You can find an example below.

Example:

#./calc_server &

CalcServer PID : 77234

#./calc_client 77234 2 + 3

The server has calculated the result of 2 + 3 as 5.00

Steps:

1. Create a new Momentics workspace named: **cst8244_lab5**
2. Create two new projects named **calc_server** and **calc_client**. Create both projects as QNX Projects > QNX Executable.
3. Download the include file "calc_message.h" (src: Brightspace) to define the typedefs to be used for sending requests and receiving responses. Only one copy of the header file should appear in your Momentics.IDE workspace. Put the header file in the calc_server project space. The client (calc_client) is to reference the header file.
4. Create the *calc_server* and *calc_client* programs.
5. Test your programs.

Deliverables

Before making the zip-file deliverable (next item), please action:

- Format your source code to make it easier for me to read. Momentics IDE will do this for you: Source -> Format

- Verify your projects build cleanly. Please action: a) Project -> Clean… and b) Project -> Build All

Prepare a zip-file archive that contains the following items:

1. Export your projects as a zip-archive file.
   Momentics IDE provides a wizard to export your project:
   File > Export… > General > Archive File > Next > Select All > Click 'Browse…" > Save As: **cst8244_lab5_*yourAlgonquinCollegeUsername*.zip** > Save > Finish
2. Take a series of screenshots of your Neutrino terminal window that captures these scenarios:
   i. A normal scenario for: +
   ii. A normal scenario for: -
   iii. A normal scenario for: x

   Note: avoid testing with '*' for multiplication, as the Neutrino command interpreter will perform file globbling (i.e. the interpreter sees '*' as wildcard).

   iv. A normal scenario for: /
   v. An abnormal scenario for server: division by 0 (zero)
   vi. An abnormal scenario for client: invalid operator
   vii. An abnormal scenario for server: server overflow
   viii. An abnormal scenario for client: no command-line arguments (expect to see usage message)
   ix. An abnormal scenario for client with no server running: what happens if you terminate calc_server, and then launch calc_client (i.e. ./calc_client 2 2 + 2)? The client is to terminate gracefully (i.e. no crash please) but with an error message.

3. A "README.txt" file reporting the status of your lab. Follow this template:

   Title {give your work a title}

   Status
   {Tell me that status of your project. Does your program meet all of the requirements of the specification? Does your program run, more importantly, does your program behave as expected? Does your program terminate unexpectantly due to a run-time error? Any missing requirements? A small paragraph is sufficient.}

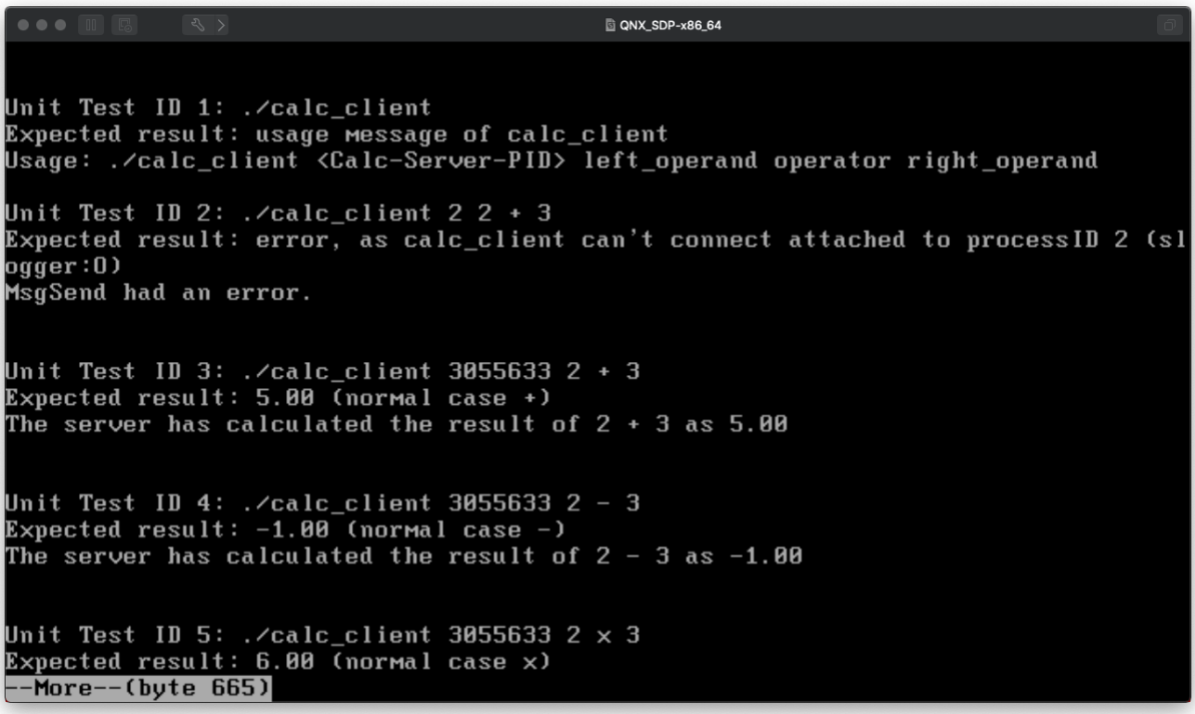
   Known Issues
   {Tell me of any known issues that you've encountered.}

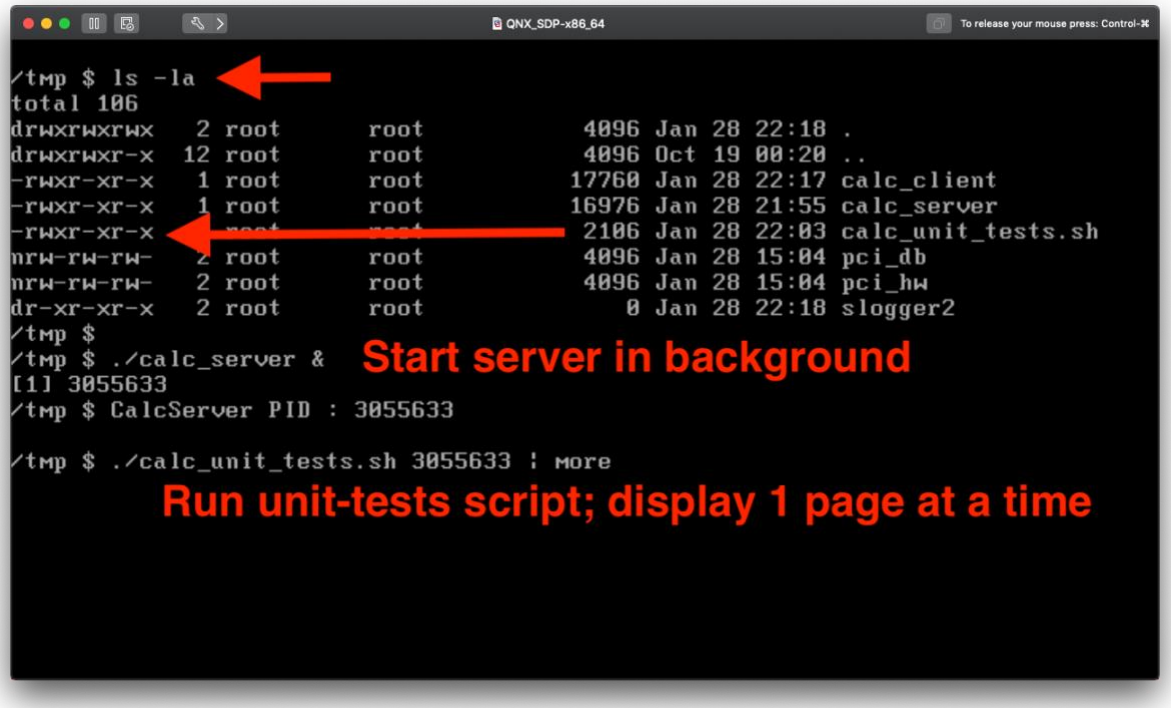
   Expected Grade
   {Tell me your expected grade.}

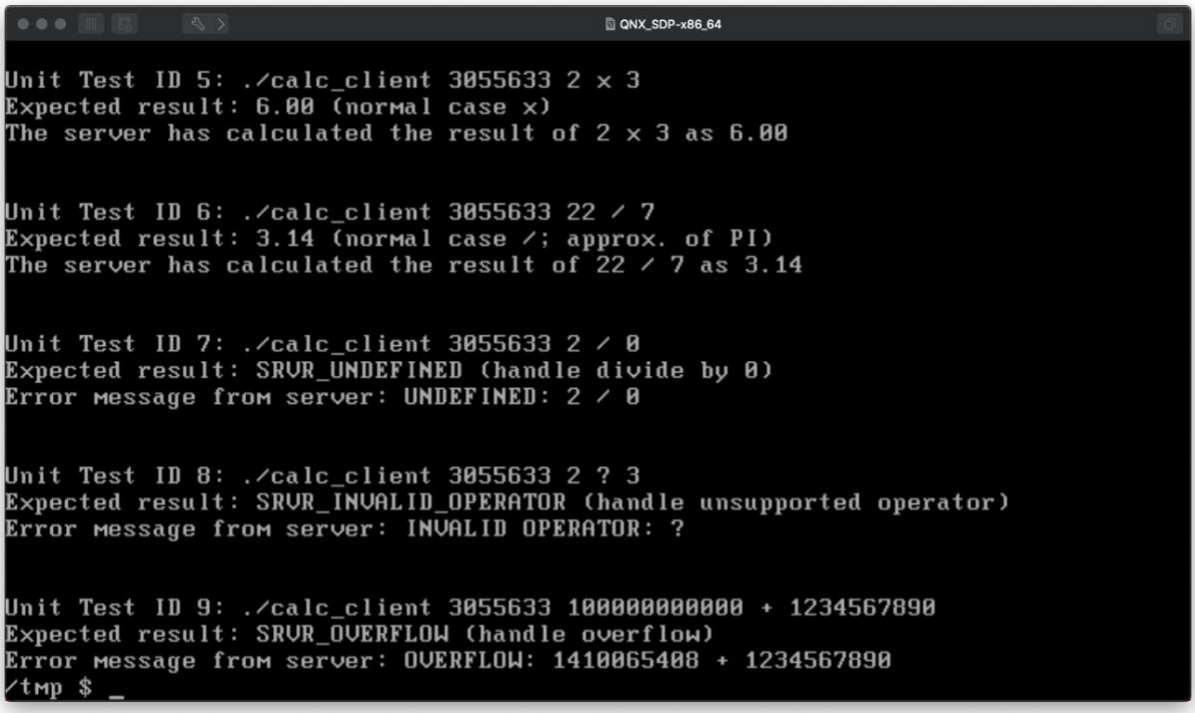
Upload your zip-file to Brightspace before the due date.

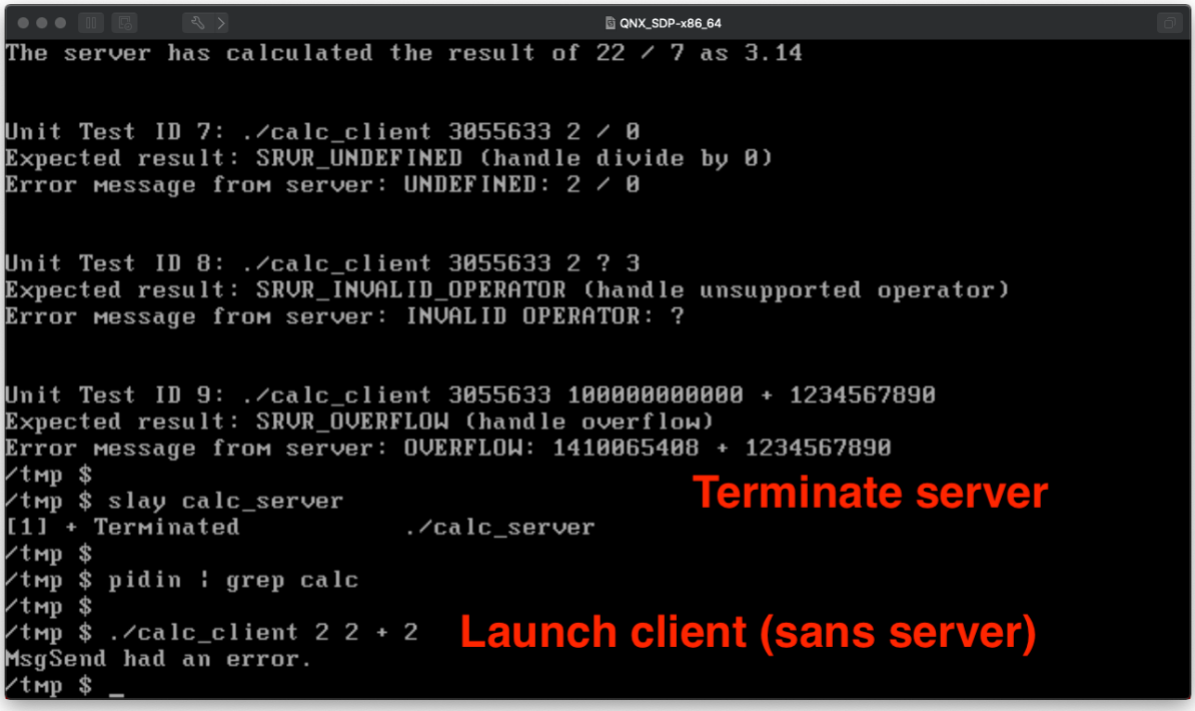
Reference Screenshot

Your screenshots are to match mine:




QNX_SDP-x86_64
To release your mouse press: Control-⌘
/tmp $ ls -la
total 106
drwxrwxrwx 2 root root 4096 Jan 28 22:18 .
drwxrwxr-x 12 root root 4096 Oct 19 00:20 ..
-rwxr-xr-x 1 root root 17760 Jan 28 22:17 calc_client
-rwxr-xr-x 1 root root 16976 Jan 28 21:55 calc_server
-rwxr-xr-x 2106 Jan 28 22:03 calc_unit_tests.sh
drw-rw-rw- 2 root root 4096 Jan 28 15:04 pci_db
drw-rw-rw- 2 root root 4096 Jan 28 15:04 pci_hw
dr-xr-xr-x 2 root root 0 Jan 28 22:18 slogger2
/tmp $
/tmp $ ./calc_server &
Start server in background
[1] 3055633
/tmp $ CalcServer PID : 3055633
/tmp $ ./calc_unit_tests.sh 3055633 | more
Run unit-tests script; display 1 page at a time


QNX_SDP-x86_64
Unit Test ID 5: ./calc_client 3055633 2 x 3
Expected result: 6.00 (normal case x)
The server has calculated the result of 2 x 3 as 6.00
Unit Test ID 6: ./calc_client 3055633 22 / 7
Expected result: 3.14 (normal case /; approx. of PI)
The server has calculated the result of 22 / 7 as 3.14
Unit Test ID 7: ./calc_client 3055633 2 / 0
Expected result: SRVR_UNDEFINED (handle divide by 0)
Error message from server: UNDEFINED: 2 / 0
Unit Test ID 8: ./calc_client 3055633 2 ? 3
Expected result: SRVR_INVALID_OPERATOR (handle unsupported operator)
Error message from server: INVALID OPERATOR: ?
Unit Test ID 9: ./calc_client 3055633 100000000000 + 1234567890
Expected result: SRVR_OVERFLOW (handle overflow)
Error message from server: OVERFLOW: 1410065408 + 1234567890
/tmp $ _



How to Run the Client and Server

Please refer to the **Resources** module on Brightspace for suggestions on how to run your client and server programs on Neutrino. My personal favourite… drag 'n drop from Momentics IDE to Neutrino ☺