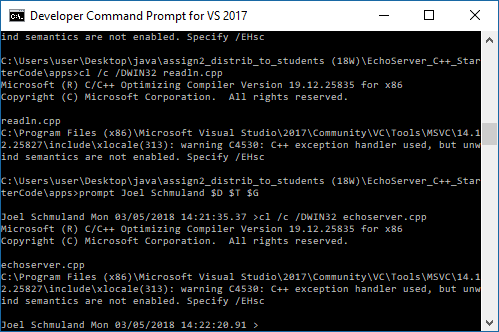
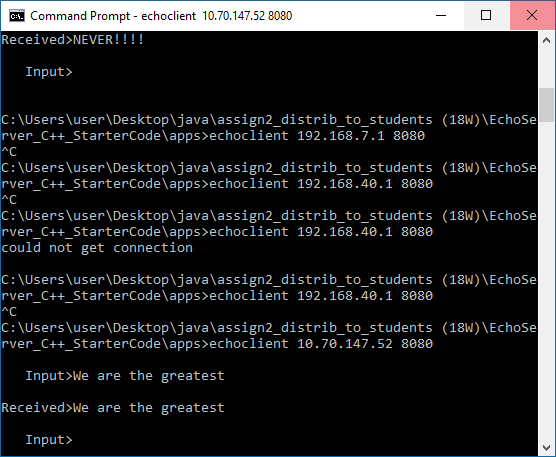
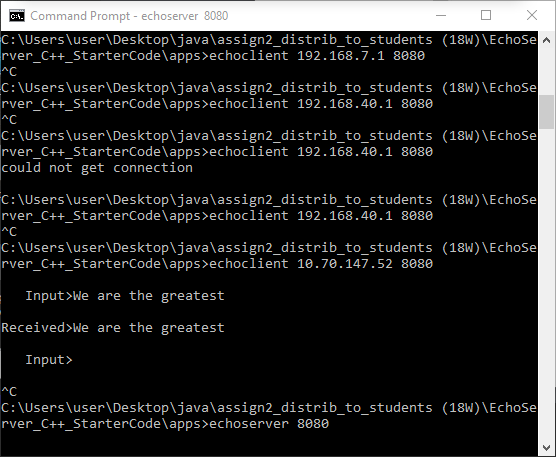
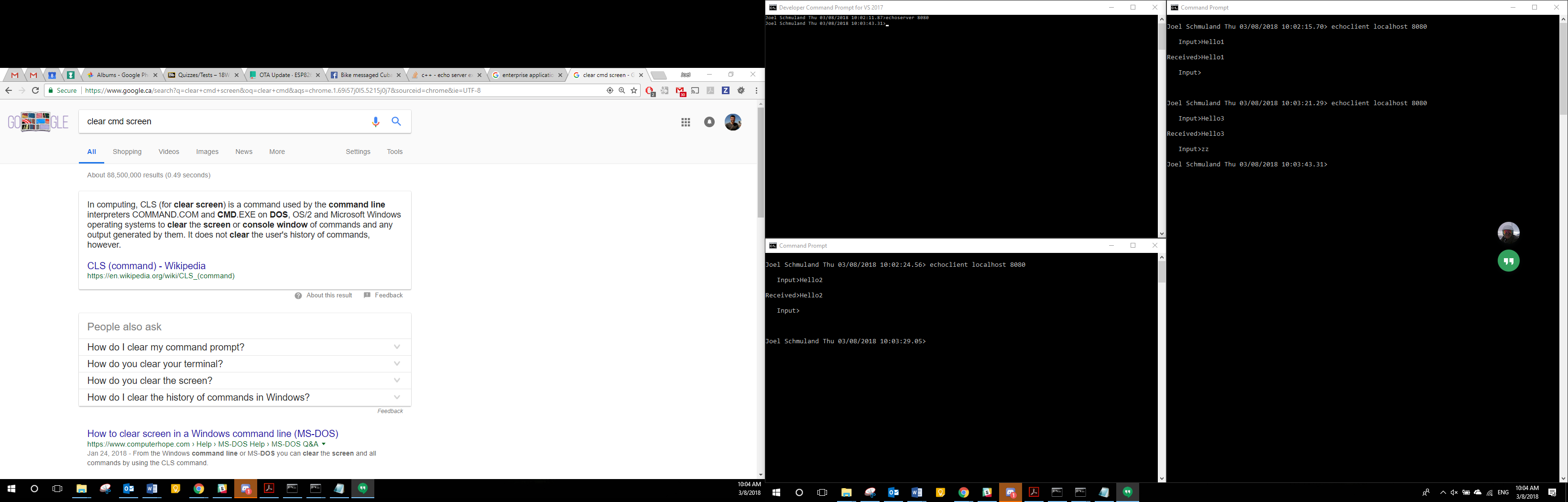
# Part A – EchoClient, EchoServer with C++

### Screen Shots

Before changing the code:  






After changing the code running on one machine:  


### Essay on C++ preprocessor, compiler, linker

The preprocessor, compiler, linker are 3 steps in getting a C++ application into a library or executable file. They go in order as stated above with the preprocessor coming first.

Its job is to take a C++ file and include the extra code that is pulled from the include statements. It spits out source code that doesn’t have any include statements but has the code needed to run the program. This is known as a “pure” C++ file [1].

Next comes the compiler, it takes the newly formed preprocessor code and produces the object file needed for the next step of the process. This next step is the linker. The linkers job is to take whatever object files are needed for the library or executable and combine them together, thus ending the workflow of creating an executable from C++ source code.

### Test Plan for the C++ program

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Description | Pre-Conditions | Post-Conditions | Test Methodology | Expected | Actual | Notes |
| First C++ Client can connect to server | Server is already running on port 8081 | First Client connects to server | Run client and server enter hello and view response | Client connects, sends “hello”, gets echo response of “hello” with additional information from server. | Matches |  |
| Second C++ Client can connect to server | Server is already running on port 8081 with first client still connected | Second client can connect but is not responsive until first client disconnects | Run server, run first client, run second client, shutdown first client | Second client should send and receive hello2 after first client is shutdown | Matches |  |
| Third C++ Client can connect to server after both first and second client disconnect | Server is already running on port 8081 with 1st and 2nd client having connected and disconnected | 3rd client connects to server | Run server, run 1st client, run 2nd client, shutdown both, run 3rd client | 3rd client should send and receive hello3 after 1st and 2nd client have shut down | Matches |  |
| Shutdown server from any client | Server is already running on port 8081 and a client is connected | Client sends signal and shuts down the server and itself | Run server, run client, send “zz” to signal the server to shut down | Both client and server shut down after “zz” is sent | Matches |  |

Test boundary conditions:

Can the C++ server handle more than one C++ client, sequentially? yes

Does the server keep running if a client shuts down? Yes

Does the server shut down when the client sends the correct signal? Yes

# Part B

### Short Answer Questions

1. What is a serialVersionUID and why can omitting it cause problems (consider cross-platform development with different Java compilers and run-times)?
2. What is a UUID? What can it be used for?
3. Research and very briefly describe each of these protocols (cite your sources): SMTP, FTP, HTTP, SOAP+XML, and REST+JSON (What does the acronym mean, in general what is it used for?).

### Test plan for the Java Program

# References

|  |  |
| --- | --- |
| [1] | R. M. Fernandes, "Stack Overflow," Stack Exchange Inc, 9 January 2017. [Online]. Available: https://stackoverflow.com/questions/6264249/how-does-the-compilation-linking-process-work. [Accessed 08 03 2018]. |