CS260 Assignment #1:
Introduction to Winsock and UDP

Task

In this assignment, you will get your first experience with socket programming by writing a very simple Windows console application. Your program will send a UDP packet to a remote server, wait for a UDP packet in response, print out the contents of that UDP packet and then exit.

Your program will need to go through all the steps outlined in the Winsock slide deck:

- Initialize Winsock
- Create a socket
- Create a remote address from the string "104.131.138.5"
- Send data over UDP
- Listen for a response
- Close the socket
- Shut down Winsock

In future assignments, you'll need to get parameters like the remote IP address from the command line. In this assignment, however, just hard-code the values: The remote IP address is 104.131.138.5, and the remote port number is 8888. The data you should send is whatever you named your program (for example, "assignment1.exe"). This value should be retrieved from argv[0] of main().

When you receive a UDP packet in response, simply print the contents of that packet to the console. Note that there's no guarantee the packet will have a null byte at the end (in fact I promise you it won't), so you will need to make sure it does before you try to print it.

Testing

The server you'll talk to for this assignment is currently running, so you should be able to test your code at any time. If you can't get a response from the server, first see if the server received your packet: In a web browser, go to http://echo.cs260.net/packets. That web page will report all the packet activity the server has seen from your IP address.

If the packet log on the web server says "not found", then the server hasn't seen any packets from you. In that case, try hitting the server with Packet Sender (http://packetsender.com). If that fails, then send me an email and I'll investigate--it's possible the error lies with the server, not your client.

Submission

Write your program as a C or C++ Visual Studio project, using the current version of VS. Once you have finished your program, make sure it compiles and runs successfully as a release build. Then zip up your entire project directory and post it to Moodle.

Grading

This is a deliberately simple assignment. From a base score of 100, you'll lose the following points for errors:

- If your program can't be compiled in release mode with a default installation of VS, you lose 100 points. Don't rely on any external tools such as cmake.
- If your program doesn't send a packet to the server, you lose 70 points.
- If the packet you send to the server is more or fewer bytes than the name of your executable file, you'll lose 10 points. Sending the entire buffer, or sending a string with its null terminator, are common mistakes you should watch out for.
- If your program doesn't receive and print out the server's complete response, you lose 30 points.
- If your program prints out the server's response but prints more or fewer bytes, you lose 10 points. Again, it's very common to allocate a buffer for the response and then fail to null-terminate it before calling printf; don't do that.
- If your program prints *anything else* out besides the received packet contents, you lose 10 points. Some of your future assignments will involve communicating over rigidly-defined protocols, and/or will be graded using automated unit tests, so you need to learn now to precisely follow specifications.

If you score 89 or fewer points, you may correct errors and resubmit once after the deadline has passed. This resubmission carries a 10 point penalty. (Correcting errors and resubmitting before the deadline is free.)

Due Date

Check Moodle for the due date of this assignment. Your submission is due before midnight on the indicated date. Submitting any time after that counts as your one resubmission.