Objective of this assignment:

- To become you familiar with developing and implementing TCP or UDP sockets.

What you need to do:

1. Implement a simple TCP Client-Server application
2. Implement a simple UDP Client-Server application
3. Collect and analyze round trip time measurements for each of the above applications.

Objective:

The objective is to implement a simple client-server application using a safe method: start from a simple **working** code for the client and the server. You must slowly and carefully *bend* (modify) little by little the client and server alternatively until you achieve your ultimate goal. You must bend and expand each piece alternatively like the way a black-smith forges iron. From time to time save your working client and server such that you can roll-back to the latest working code in case of problems.

For this programming assignment, you are advised to start from the simple echo client and server to implement a very simple application.

Part I: TCP "Devowelize" Client-Server

Implement the following Client-Server application that will use two programs: a client program myFirstTCPClient.java and myFirstTCPServer.java

a) **Client: myFirstTCPClient.java**

This program must take three **command arguments**: a hostname H, a port number P, and a string S where the hostname H is a name or a decimal dotted-quad IP address of the server Sv, the port number P is any valid port number where the server Sv is bound to, and the string S is any sentence of at most 255 characters.

This program must:

1) Create a TCP client socket connected with the server Sv running on the machine with hostname (or IP address) h bound to Port number P.

2) Perform the following actions:

 i) Send the sentence S to the server Sv

 ii) Receive the response from the server

 iii) Measure the duration between the time when the sentence S was sent and the time a response was received.

 iv) Display the following information: the message received and the time expressed in milliseconds.

 v) Collect the round trip time.

To implement the client myFirstTCPClient.java, you should consider start with the program *TCPEchoClient.java* (provided on Canvas with this programming assignment). Do not forget to change the name of the class inside the program *TCPEchoClient.java*.

b) **Server: myFirstTCPServer.java**

This program must take one argument: a port number P. The port number P is any valid port number.

This program must:

1) Create a TCP server socket

2) Wait for a client to connect, receive a message, display it with the IP address and port # of the client, "*devowelize*" the message, display the *devowelized* message, and echo back the *devowelized* message. **Devowelize** a sentence S means to remove any vowel (*a, e, i, o, u*) in the sentence S. Dewolizing the sentence "Hello World!" yields " Hll Wrld!".

To implement the server myFirstTCPServer.java, you should consider start with the program *TCPEchoServer.java* (provided on Canvas with this programming assignment). Do not forget to change the name of the class inside the program *TCPEchoServer.java*.

Part II: UDP "Devowelize" Client-Server

Repeat Part I using **UDP** sockets. Call the client and server programs myFirstUDPClient.java and myFirstUDPServer.java, respectively.

To implement the server (respectively, client) myFirstUDPServer.java (respectively, myFirstUDPClient.java), you should consider start with the program *UDPEchoServer.java* (respectively, *UDPEchoClienTimeout.java*) (provided on Canvas with this programming assignment). Do not forget to change the name of the class inside the program.

Data collection and analysis

For **each** application (UPD and TCP), report separately the min, average, and max round trip time.

Report

- Write a report that will report your results. The report should not exceed half a page.
- In addition, your report must contain the following information:
    - o whether the programs work or not (this must be just ONE sentence)
    - o the directions to compile and execute your program

What you need to turn in:
- Electronic copy of your source programs (separately standalone: i.e., not in a zipped folder or archive)
- Electronic copy of the report (including your answers) (standalone). Submit the file as a Microsoft Word or PDF file.

Grading
1) TCP client is worth 20% if it works well: communicates with **YOUR** server (designed by your team).
2) TCP client is worth 5% more if it works well with a working server from any **other** team.

All other server and clients (TCP server, UDP client, and UDP server) will be graded the same as the TCP client (20% + 5%).