CSUS
COLLEGE OF ENGINEERING AND COMPUTER SCIENCE
Department of Computer Science

CSc/CpE 138 Computer Networks and Internets
Spring 2017
Ghansah

Programming Assignment #1 (Introduction to Socket Programming)
THE FOLLOWING PROBLEM REQUIRES DEMONSTRATION. Due:

THE GRADER MIGHT TEST YOUR SUBMISSION FOR ACCURACY. IF NECESSARY, DEMONSTRATION WILL BE SCHEDULED.

Given the size of the class and the requirement for demonstration you should work in groups of no more than two students. There will be no exceptions. That is, I will not accept individual work. You must submit a joint report indicating names of students on the first page as well as who was responsible for what. It is assumed that subsequent programming assignments will use the same grouping.

NOTE: Although the examples discussed in the text are in Python your submission can be in C or Java if you choose do so with the caveat that there is more help (see below) at the textbook authors' website if you do it in Python.

Socket Programming Assignment 1: Echo Client/Server

Note: This is the first of a series of programming assignments in the text book that will be assigned in the course of the semester. Students can find full details of these assignments, as well as important snippets of the Python code, at the Web site for the text book. http://www.awl.com/kurose-ross. For help with Python, google is your friend. For help with the following Echo assignment, google is your best friend with respect to Python syntax, etc.

Assignment 1 Echo Client/Server

Part a) Echo Client/Server

In this assignment, you will develop a modification of the simple Echo client/server Python implementation that we discussed in class. Recall that the client receives data typed on a keyboard, sends the data to the server, the

server converts it to upper case and returns the result to the client. The client then displays it on the screen. In the modified version, you should let the client and server stay in a loop until the Client submits a ‘quit’ command on a separate line then the program ends.
Your job is to complete the code with the modifications, run the code, for both client and server and test it for both UDP and TCP.

Part b) Modified Chat-like Application
Similar to Part a) except that each machine, A or B is both a client and a server. That is, a user on machine A types something to B which converts it to upper case and echoes it back to A. A similar thing happens if someone on B types something to A. This process continues until one user types ‘Quit’ on a separate line. For simplicity, sequence does not matter much in this application. There are multiple options for the configuration. You can 1) Use a pure peer-to-peer system where each machine is both a client and a server operating in a multi-threaded fashion; 2) Let A and B clients to the third party server that coordinates client activities; 3), etc.

In all cases the transport protocol you choose is up to you but just pick one (UDP or TCP).

DELIVERABLE: Objectives, Design, Documented code, and answers to all relevant questions (if any).

DEMONSTRATION will test among other things, whether you understand what is going on. Your grade on this will be based on preparation, understanding, and answers to questions, including source code.

SUBMISSION: Electronic Submission Only (**IN SACCT according to the format given in the course syllabus**)! NOTE: IF YOU HAVE MULTIPLE FILES YOU CAN PUT THEM IN A DIRECTORY AND SUBMIT THE Zipped Directory with the name according to the format specified in the course syllabus.