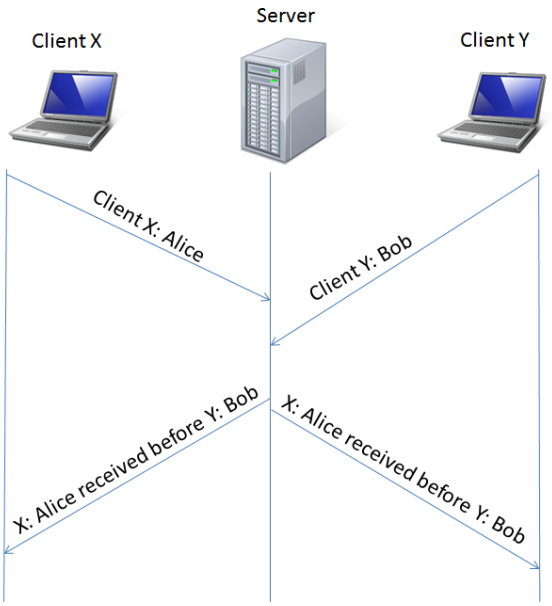
# CS 428/528

## Instructor: Anand Seetharam

## Programming Assignment 2: Socket Programming

In this assignment, you'll write a server and client program. There are two clients, X and Y (both essentially identical) that will communicate with a server. Clients X and Y will each open a TCP socket to your server and send a message to your server. The message contains the name of the client followed by your name (e.g., "Client X: Alice", "Client Y: Bob").

The server will accept connections from both clients and after it has received messages from both X and Y will print their messages and then send an acknowledgment back to your clients. The acknowledgment from the server should contain the sequence in which the client messages were received ("X: Alice received before Y: Bob", or "Y: Bob received before X: Alice"). After the server sends out this message it should output a message saying - "Sent acknowledgment to both X and Y". Your server can then terminate. Once your clients receive the message from the server, they should print the message that they sent to the server, followed by the reply received from the server. The following figure explains the problem.




Your program should print out the messages sent by the client at both the client and server and vice versa for messages sent by the server.

**Extra Credit: (10%)**

You can modify the above server-client to create a simple chat service. Clients X and Y can only chat through the server. For example, every message that client X sends to the server, the server relays to client Y and vice versa. When a client (say X) wants to exit the chat service it sends a "Bye" message. When a server sees a "Bye" message, it relays this message to Y and then terminates the connection to both clients. Each client (say X) should output the messages sent by it and those received from Y. As this is a chat service the number/content of messages exchanged is not fixed. So your clients should have the capability to accept inputs (which are the content of the messages) from the keyboard.

## What programming language to choose?

You can code in any language of your choice – C, C++, Java and Python. If you are using any other programming language please talk to me first. Here are a few tips for coding in any of these languages.

### *Programming in Java*

Java encapsulates the concept of a client-side connection-oriented (TCP) socket with the class, Socket. Java encapsulates the concept of a server-side connection-oriented socket with the class, ServerSocket. You'll need to use the accept() and close(0) methods of this class.

If you're interested in a quick Java tutorial targeted specifically at socket programming, check out "Socket Programming in Java: a tutorial," by Q. Mahmoud, Javaworld, Dec. 1996, http://www.javaworld.com/article/2077322/core-java/sockets-programming-in-java-a-tutorial.html .

### *Programming in C*

If you choose to code in C, you will need to learn the use of system calls such as socket(), bind(), listen(), accept(), close(). Here is a url for socket programming http://www.lowtek.com/sockets/

### *Programming in Python*

The socket programming code in your book is written in python. An online Python socket tutorial is https://docs.python.org/2/howto/sockets.html

## What to hand in?

1. Submit code with in-line documentation.

2. Run your code on your local machine as well as on ‘remote’. A design document outlining the decisions you made and also sample outputs (screenshots), which show that you program works correctly. Please include screenshots that show that your code runs correctly on your machine as well as on remote. If you had to make any changes to run your code on remote, please mention that in your design document. A brief description of cases where you code might fail and possible ways of improving your program.

3. Execute the code to me to demonstrate that it works correctly.

## Grading Criteria

In-line documentation - 20 points

Code compiles and executes correctly – 60 points

Design document – 20 points