Lynne Coblammers
EECS 678
Lab 4
Sockets

1. Briefly describe the design of the program. How does your program control when the client runs and when the server runs?

Both the client and server run concurrently. The server establishes a handshake socket and then listens for connections. When the client starts up, it connects to the server's handshake socket. The server accepts the connection and creates a new session socket, which it returns to the client. The client and server are then connected. The server reads from the socket anything that the client sends until the clients socket is closed. Each time it reads a line, it prints it, transforms it, then writes to the socket. The client, alternatively, prints a string, writes it to the socket, and reads what the server sends back so it can be printed. This continues until the entire array of strings has been sent to the server, at which time the client closes the socket. Finally the server closes its two sockets.

2. What is the purpose of the handshake socket? Why not have the server create and bind session sockets that clients may connect to directly?

The handshake socket allows for two things. First, it allows clients to know a single location to connect to. Rather than having to store and try multiple session sockets, they store a single handshake socket and always connect to that. Second, it allows the server to create as many sessions as it needs without having to know in advance how many session sockets to create. Until someone connects, it only has to maintain a single socket, and can add sockets each time a connection is made.

3. For the simple / client server program, we chose to use sockets instead of pipes to send messages between the client and server. Why are sockets preferred over pipes for this program? Give at least two reasons.

One reasons sockets are preferred for this application is that they provide bidirectional communication, as opposed to unidirectional communication from pipes. To get the same functionality, two pipes would be needed rather than a single socket, one going from server to client, and one from client to server. Another reason sockets are preferred is that they allow for connection between separate computers. Since servers are often at a remote location, separate from the clients, servers would not be able to communicate with the clients via pipes, which only work on a single computer. Instead, they can use sockets to connect over the internet.