Server:

When a server starts up, it creates a socket and binds and listens to the ip/port combination specified through the command line. A new thread is started to handle incoming connections or messages received on sockets created by the server. When a server receives a connection on this socket, it creates a socket, and waits to receive a Register message from the client. When received, the client is added to the directory, which is then sent to all sockets currently opened. When a client wants to disconnect, an EXIT message is sent containing the IP and address of the exiting client. The server then closes the socket that corresponds to this client.

Client:

When a client starts up, it creates a socket to connect to the ip/port combination specified through the command line. It also creates a socket to connect to the server-ip/server-port combinations specified through the command line. It then sends a Register message through the socket to the server. A second thread is then started up to connect and receive messages through other sockets. When a directory is received by a client, it updates its own directory to include additional ips specified by the server's dir, not including its own ip and port. The client can then connect to any ip in the directory by opening a socket to connect to it and send status messages giving char*s to the other client specified.

Client Functions:

void acceptAndReadMessages() – loops endlessly waiting for incoming connections, or reading/acting on messages received by sockets.

StringArray* getIps() – returns the ips currently in the clients directory

void sendMessage(char* ip, const char* message) – sends a status message containing message to the ip/port combination in ip.

virtual void closeSockets() – closes all sockets connected to the client

void waitForConnectionsAndRead() - starts the acceptAndReadMessages method on a new thread

Server Functions:

The server contains the functions listed above, minus getIps.

broadCastToAll(Message* mess) – sends the message to all sockets connected to the server.