Structure Definitions for Client and Server Structures

Data Structures

There were three main data structures created on the server side. All are listed below

- `Message_node`
- `list`
- `Client`

Message_node structure outlines how the individual messages themselves are stored and handled. Each Message_node contains a message, being the text of an actual message, a channel_id being the channel which the message belongs in, list count which represents which number this message is in the linked list. Channel_count which represents which number this message is in the channel since the start of the channel, next_message which is a pointer to the next message which is sent to the server and next_chan_mes which points to the next message which is in this particular channel.

The second structure List is a list of the messages pointing to the last message sent to the server and the last message sent to any given channel. With two message_nodes inside storing the last message and channel_list_end[MAX_CHANNELS] which records the last message sent to each channel as a pointer to each message

The Client structure is for storing the clients information. This includes socket which stores the socket for the client, client_id which stores the ID for the client given by the server, connected which when set to 1, client is connected, if it is set to 0, client is disconnected and should be reset. Two message nodes, first_message_since_joining[MAX_CHANNELS] which points to the first message sent to a channel the client is subscribed to since joining the server and read_channel_message[MAX_CHANNELS] which is an array of pointers to the last read message on channels the user is subscribed to.

Compiling

In order to compile the Client.c and Server.c files into programs, change directories into the local folder with the files on them and run make. If the Makefile fails, the glocal compiler was used with no flags for both programs:

```
gcc -o Server Server.c
gcc -o Client Client.c
```