Exercise: Network "In/Out Board" Using Sockets

In a distributed environment, it is difficult to answer the question "who is in the office today?" This exercise involves producing a simple client/server network tool to alleviate the problem.

This exercise involves producing two separate Java sockets applications:
1. Client application
2. Server application

The exercise also requires that you understand and implement an application-level protocol so that these two applications can operate with each other.

To make the development of a client easier for you, there is a server application that you can use to test against. As the exercise progresses, you will replace that server with one of your own making.

1. Protocol

The client and server communicate via a simple protocol, as follows:

	Client Sends	Server Responds
Mark 'fred' in	Ifred where I is a single char and fred is a UTF stream	OK or ERROR: + return from Exception.getMessage (), as a UTF stream
Mark 'fred' out	Ofred where I is a single char and fred is a UTF stream	OK or ERROR: + return from Exception.getMessage (), as a UTF stream
List in/out board entries	The single char L	First: count (int) [status/who (UTF stream, 1st character is status, rest is who)] repeated count times. Then: OK or ERROR: + return from Exception.getMessage (), as a UTF stream

2. Client Application

The InOutClient implements the user portion of the client/server system. It is operated from the command-line only and takes either three or four parameters.

Synopsis

```
java InOutClient server-address port [user] command/status
```

Example

```
java InOutClient staffserver0 4567 Bob In
```

InOutClient connects to the address staffserver0 using port 4567 and tell the InOutServer running there that the user Bob is now in the office.

```
java InOutClient staffserver0 4567 Bob Out
```

Mark that the user Bob is now out of the office.

```
java InOutClient staffserver0 4567 List
```

Give a list of all the current users and their dispositions.

3. Server application

The InOutServer implements the server portion of the client/server system. It is operated from the command-line only and takes a single parameter.

Synopsis

```
java InOutServer port
```

Example

```
java InOutServer 4567
```

Initiates the service on port 4567 (the examples above assume that the machine executing this server is named 'staffserver0').