COMP 4958: Lab 5

In this lab, you are asked to implement a globally-registered GenServer that broadcasts messages to subscriber processes. In order to be able to handle external clients (ones not implemented in Elixir), we need helper processes that use sockets to "liaise" the external clients with the broadcast server.

Create a mix project named `Broadcast`. You need to implement at least 2 modules:

- `Broadcast.Server` implements a GenServer that provides at least the following functions in its client API
  - `start()`: starts & (globally) registers the server; the name of the server is the module name
  - `subscribe(pid)`: adds the process with the specified PID to the subscriber list
  - `unsubscribe(pid)`: removes the process with the specified PID from the subscriber list
  - `broadcast(message)`: broadcasts `message` to all processes in its subscriber list
- `Broadcast.Helper` implements a regular process that acts as the "middle man" between the server & external clients. Its purpose is to "forward" messages between its clients & the broadcast server. It has a `start` function that takes a port number (defaults to 3333) that it listens for TCP connections & spawns a process to handle each connected client. This means that each connected client has an associated helper process. Messages from the client is forwarded (by the helper process) to the broadcast server which then broadcasts it to all subscribed helper processes. The helper process then sends the message to its client. Note that a helper process will need to subscribe to & unsubscribe from the broadcast server at the appropriate time. Note also that a helper process may run on a different node than the server.

You also need to implement a suitable client in Java. Basically, this program needs to connect to a broadcast helper (via a socket), sends user input to the helper & reads & displays messages from the helper.