Summary of Project

The project is a multi-threaded client-server application. The server hosts a library of movies, actors, writers, producers, directors, and movie genres. In theory, the client is merely a curious layman who wishes to browse information about movies, but in the project as it currently stands, the client also acts as an administrator of the server-library and can add items to the library.

Both applications are written in object-oriented C++, and they use sockets with the TCP protocol to send messages to one another.

The code is pretty clear and well commented, so a more intimate knowledge of how the program works can be attained by perusing the code files.

How to Run

First of all, **the code only runs on Linux.** The thread and socket libraries that it uses are native to Linux and will not work on Windows or Mac. So assuming we are running on a Linux system, let's continue…

Download the project so that you have two separate directories on your computer: Server and Client. Open a terminal and `cd` to the Server directory. Type `'make'` and hit enter.

The Server application is now compiled and there should be an executable file named 'commerce_server' in your Server directory. (If this part did not work, make sure you have g++ compiler installed.)

This application takes 1 argument in order to run, which is the number of the port from which it will communicate with the client. So now type `'./commerce_server <port number>'` and hit enter. `<port number>` should be the number of a free port on your computer. There are tools you can use to find free ports, but if you just try a number between 6000 and 8000, it should work just fine.

If all has worked out, the server is now running and waiting for clients to connect to it!

Now, we need to run the client. Open a terminal and `cd` to the Client directory. Type `'make'` and hit enter.

Again, there should now be an executable file in the Client directory, this time called 'commerce_client'.

The client application takes two arguments – the IP address of the server and the port number to connect to the server. So now type `'./commerce_client <ip address> <port number>'` and hit enter. If you're running both the Client and the Server on the same computer, use 127.0.0.1 as the IP address. **And make sure that `<port number>` here is the same port number that you entered when running the Server.**

If all has worked out, your client and server are now connected. You can start entering commands at the client side in order to use the program. Note: it's a multi-threaded server, so you can run multiple instances of the Client application and they will all connect to the Server in parallel. All of the clients interact with the same server library so changes made by one client can be seen by the other clients.

How To Use

The project unfortunately does not have a convenient user-friendly interface yet, so entering the commands is a bit tedious. To help, I've included a file 'input_commands' with some sample commands that you can input. Nevertheless, I'll explain all of the different commands and how to enter them. There are 15 total commands, plus an "exit" command. All of the non-exit commands begin with a number denoting the type of command (1 – 15).

In all of the following commands, only the last field may contain spaces. The other fields may not. In addition, any command that doesn't print anything prints `Success` if it executes successfully and `Failure` if not.

1.) Add a new movie to the library.
```
1 [id ][name ][length ][year ][rating ][description]
```

2.) Add a new professional to the library.
```
2 [type ][id ][age ][job-description ][gender ][name]
```
For the "type" field, there are four possible types for professionals. 0 = director, 1 = actor, 2 = writer, 3 = producer.

3.) Add an existing professional to an existing movie.
```
3 [movie-id ][professional-id]
```

4.) Add a genre to a movie.
```
4 [movie-id ][genre]
```

5.) Define the sort type for the list of professionals that work on a movie.
```
5 [movie-id ][sort-type]
```
The sort type can be 1, 2, or 3. 1 = by ID number, 2 = by age, 3 = by number of movies that the professional is listed in. The default sort type is 1, meaning that if this command is not used, the movie's professionals will be sorted by ID number.

6.) Print the professionals of a movie.
```
6 [movie-id]
```

7.) Print movie details.
```
7 [movie-id]
```

8.) Combine movies.
```
8 [movie-id],...,[movie-id]
```
This command takes at least two movie ID's and combines all of the given movies together. Exact details of how this works can be found in the program comments. The ID of the new resulting movie is the concatenation of all of the ID's of the combining movies.

9.) Print all of the movies that a given professional works on.
```
9 [pro-id]
```

10.) Delete a movie.
```
10 [movie-id]
```

11.) Delete a professional.

```
11 [professional-id]
```

12.) Remove a professional from a specific movie (professional will still exist in the general library).

```
12 [movie-id] [professional-id]
```

13.) Print all the movies.

```
13
```

14.) Print all the professionals.

```
14
```

15.) Print all the movies of a given genre.

```
15 [genre]
```

16.) Exit.

```
-1
```