

Figure 1

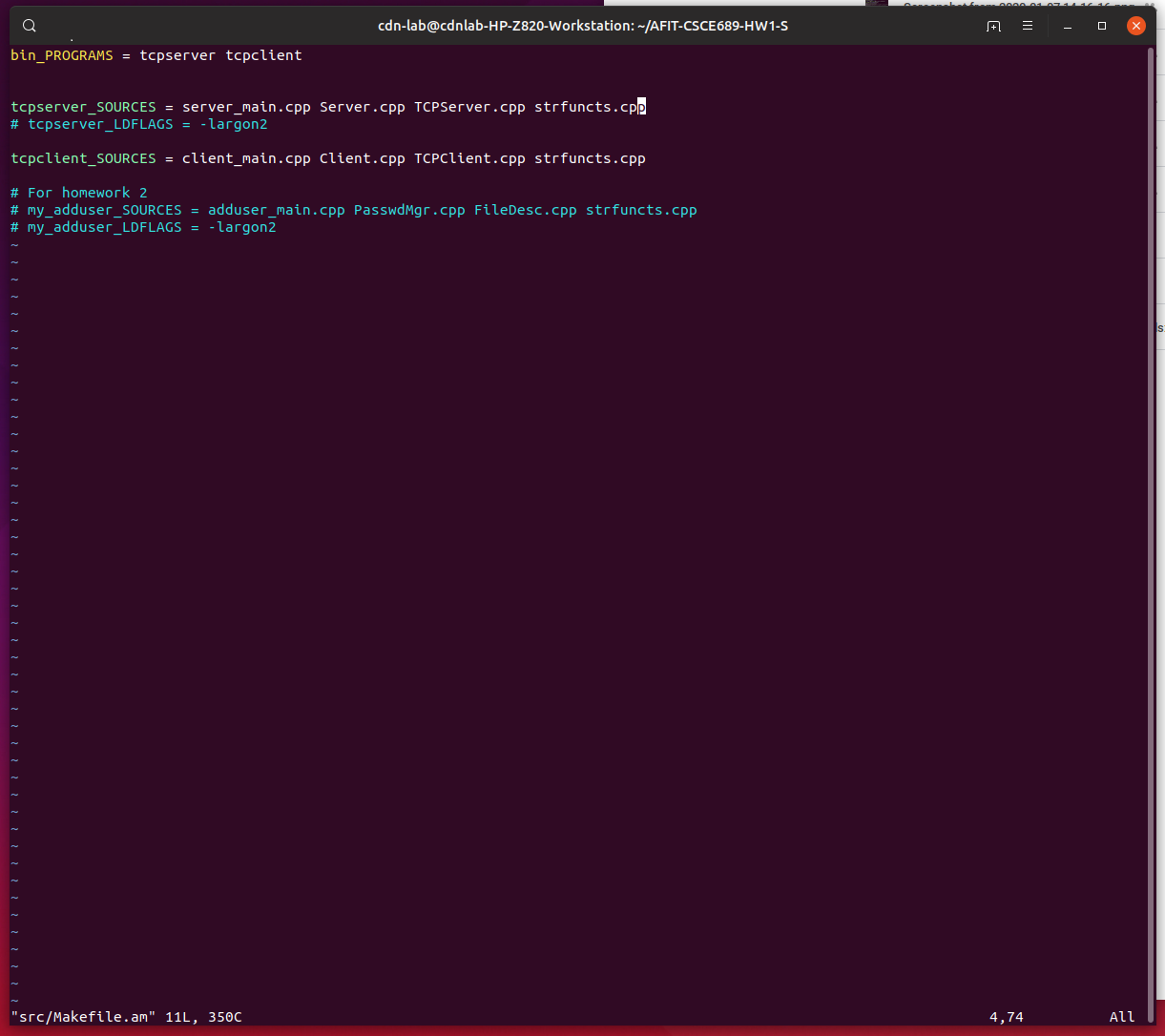
In the attached image, I first cloned Homework 1.

You can see in the homework’s root directory, it has a number of files. Most of them are unnecessary for your homework but git and the autotools complain if it doesn’t see them. Here’s a list:

* README – some info on the homework is written here
* INSTALL, AUTHORS, COPYING, ChangeLog, NEWS – Don’t worry about these
* Configure.ac – the main config file to help us build the makefiles we need (you shouldn’t need to touch this much)
* Makefile.am – Instructions to automake on how to generate the makefiles. Only includes one line that directs it to the src directory
* src directory – stores our .cpp source code files
* include directory – stores our header files
* m4 directory – will store any autotools macros we download (later homework, for now, stays empty)
* build-aux directory – will store scripts required by the autotools
* src/Makefile.am – stores config information automake needs to generate the Makefiles

So now, notice I type the command autoreconf and get some errors. Why? Because autoreconf needs some scripts in the build-aux directory to run properly. We can get it to automatically install these scripts with the command “autoreconf –i".

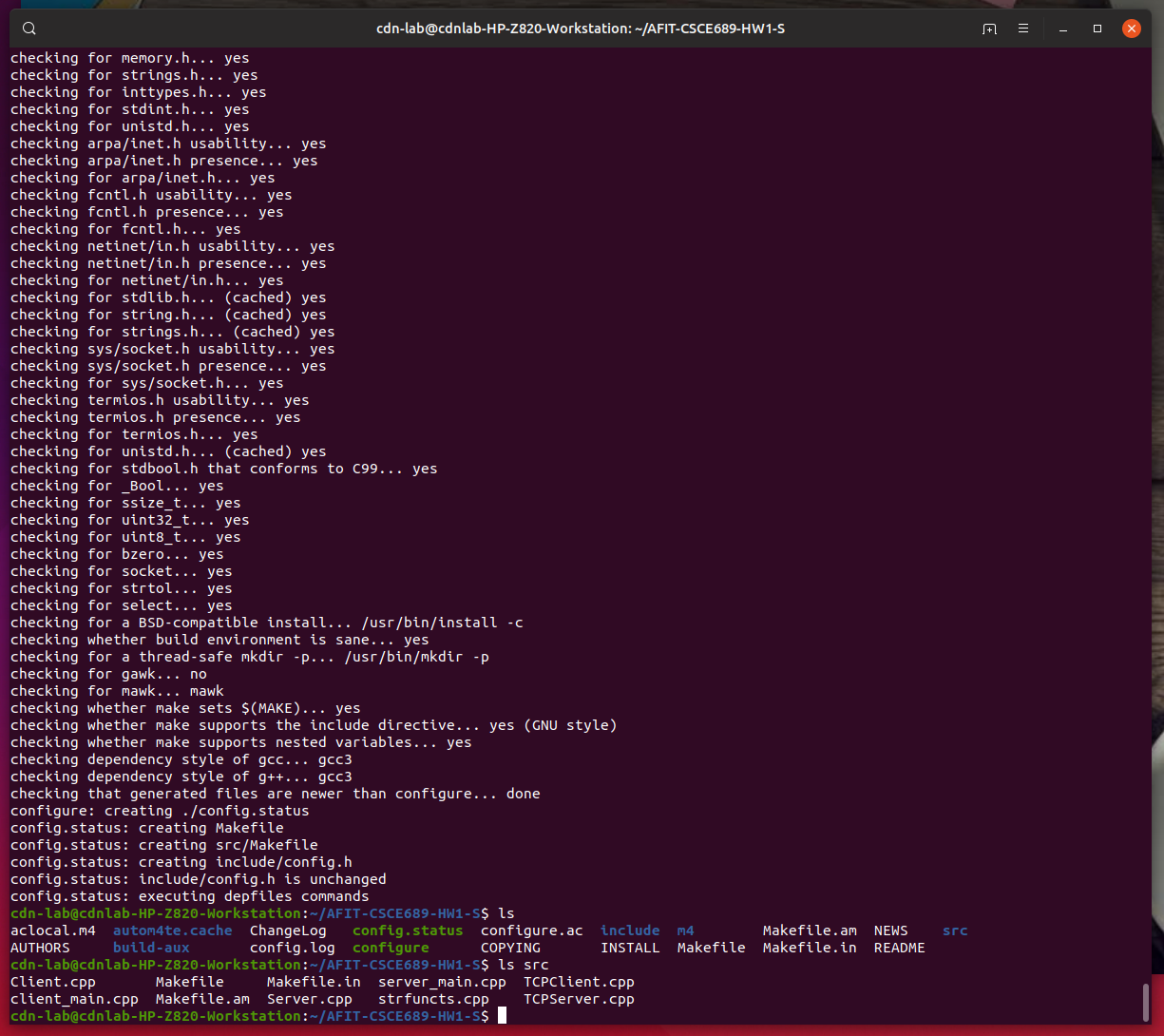
Now you’ll see that there’s a few new files in the root and src directory. Most importantly, there’s a “configure” script (green means executable) there. Also, in the src directory, you’ll see the source files as well as the Makefile.am (you should edit this) and Makefile.in (automatically generated). Before we run the configure script, let’s take a quick look at src/Makefile.am (next page).



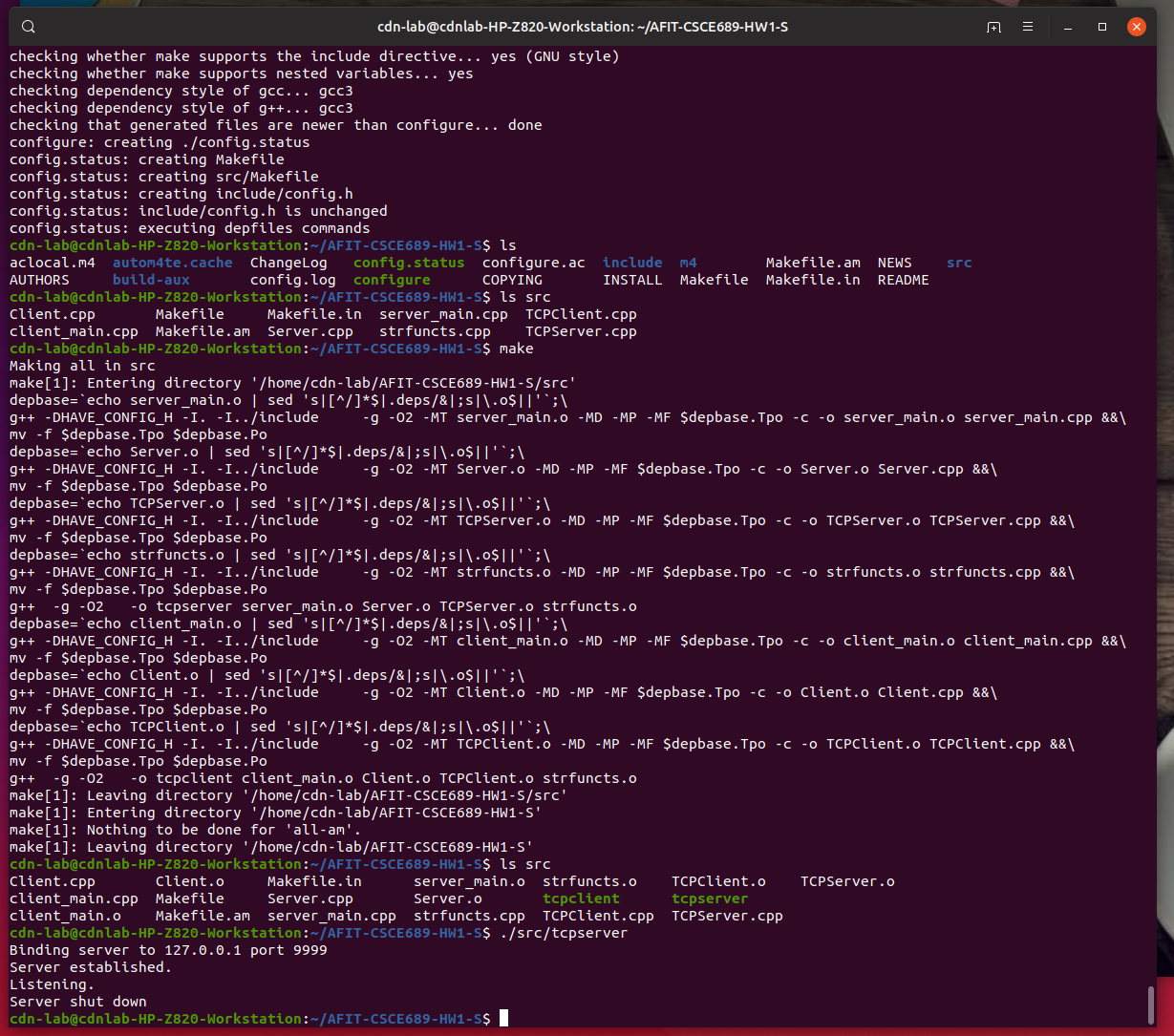
This src/Makefile.am file is where we define the binaries we want to produce. The first line is bin\_PROGRAMS where we are telling it to generate two binaries. The first called tcpserver and the next tcpclient. Next, we tell it what source files will be in each binary with the format <binaryname>\_SOURCES. So tcpserver needs the source server\_main.cpp, Server.cpp, TCPServer.cpp and strfuncts.cpp. If you create additional objects/cpp files, you’ll want to add these here.

Don’t worry about the rest for now. They are commented out and will be used in Homework 2.

Now, let’s run configure (next page).



The configure script checks to make sure all required libraries and headers are on your machine and, if there are no problems, generates a few files. Notice there is now a Makefile and src/Makefile. These are the instructions our compiler needs to compile our code (next page).



Now we type “make” and it compiles our source code. The compiler has three primary steps—preprocessing, compiling and linking. The preprocessor converts all the defines into actual values and feeds the code to the compiler. The compiler generates .o binary files, one for each .cpp file you have. The linker links the .o files into the binary. You now see there are two binary files in the src directory, tcpclient and tcpserver. They don’t anything useful right now because you need to add your code!

Good luck! See me if there are more questions.