Initial Makefile Requirements

This is a general outline of what should be done to demonstrate you have a working knowledge of makefiles and to start you off for project 1. You will create the folders before writing your makefile. If you feel that you wish to use a different directory structure, go ahead; but you will have to defend why you changed it when you pass off.

1) Create a directory with six subdirectories { bin, lib, inc src, obj, utils }

    a) **src** is for .cpp files

    b) **inc** is for .h files

    c) **bin** is for exe files

    d) **lib** is for .a and .so files (static and shared library files)

    e) **obj** is for .o files created by your project

    f) **utils** has the CS240 utilities files in it.

        i. this will have a **src**, **include**, and **obj** folder for .cpp, .h, and .o files

        ii. these files will be compiled and built into a library file in your lib folder

2) Create a makefile with these four pseudo-targets { bin, test, lib, clean }

    a) bin creates an executable (linked with the library file)

        i. for this assignment, the executable can simply print “hello make”

        ii. this will depend on lib

        iii. Later, you will implement your project main() with this file

        iv. This target does **not** run the executable

    b) test creates a different executable (linked with the library file)

        i. for this assignment, this executable simply prints “hello make test”

        ii. this target will depend on bin (which depends on lib)

        iii. Later, you will implement testing main() with this file

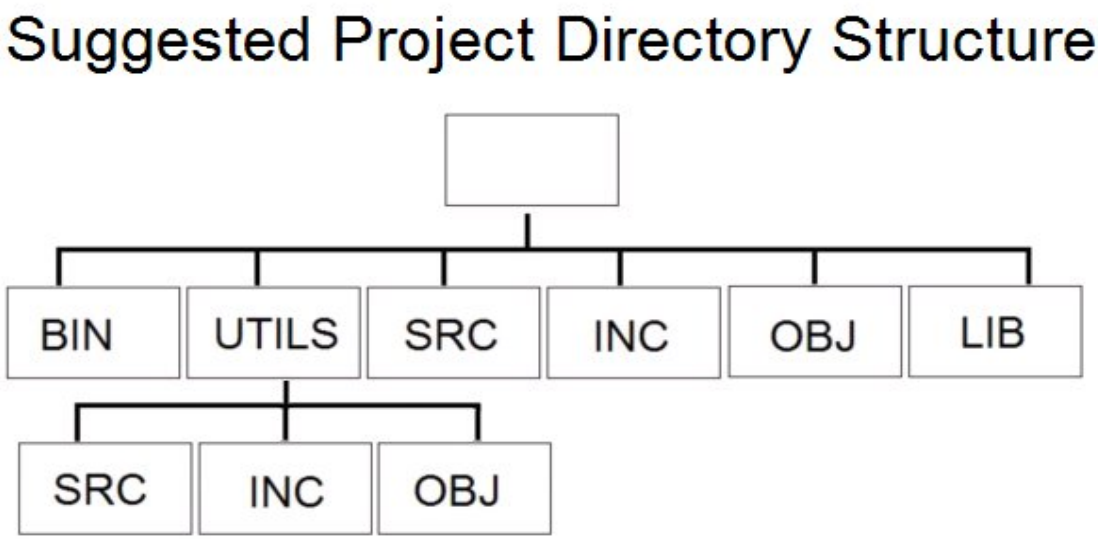
        iv. This target runs the test executable

    c) lib creates a static library (.a file) of all of the CS240 Utilities objects.

i. this target depends on each object file that will be created from the CS240 Utilities.

d) clean will delete all of the files in your bin, obj, and lib folders

i. use the –f flag with the rm command to get rid of the output (if no files were deleted)



In order to see if you are ready to pass off, follow these steps:

1. You must show your directory structure is either identical to the one we propose, or explain why you used a different structure.
2. Type “make clean”
   i. This will remove all of the .o, .a, and executable files in your directory structure.
3. Type “make lib”
   i. This will create:
      utils/obj/CommandRunner.o
      utils/obj/FileInputStream.o
      utils/obj/FileSystem.o
      utils/obj/HTTPInputStream.o
      utils/obj/StringUtil.o
      utils/obj/URLConnection.o
      lib/<the name of the utils library file>.a
4. Type “make lib”
   i. Make will tell you “There is nothing to do for lib”
5. Type “make clean”
6. Type “make bin”
   i. This will create the library (identical to step 3.i)
   ii. This will create an executable (such as bin/make240)
   iii. This executable is linked to the library
7. Type “make clean”
8. Type “make lib”
   i. Only the utils.o files and the library is created at this step
9. Type “make bin”
   i. Only the executable is created at this step

10. Type “make bin”
    i. Make will tell you “There is nothing to do for bin”
11. Type “make test”
    i. A new executable (such as bin/make240test) is created
    ii. This executable is also linked to the utils library
    iii. The test executable is run
12. Type “make test”
    i. Only the test executable is run at this point