

Lab 1

4 points

Linux, Command Line Entry and Simple C Programming

This lab is intended to provide an opportunity for students to become familiar with using an editor on Linux and the ability to run a C program on the CCC machines. Working in the lab provides ready assistance from the TAs and SAs.

Lab 1 Assignment

0. Prior to coming to the lab read and write down a solution to Exercise 4.9 in the D&D text.
1. Introduce yourself to the lab assistants. Remember, they provide input to your subjective points and they will grade your labs and programs!
2. Open a Windows session.
3. Log on onto a CCC Linux machine in your own account.
4. Create your own directory for the course (e.g. mkdir CS2303).
5. Use an editor (e.g., emacs or vi) to create a C program for your solution in step 0.
6. Use the gcc compiler to create an executable program that satisfies Exercise 4.9.
7. Test the program under several sets of test data.
8. Create a README file that contains any useful information to assist in the grading of your lab program.
9. Create a tarred file* that contains the source file and the README file.
10. Use the Unix version of the 'turnin'** to turn-in the tarred file. [The deadline for all lab turn-ins is 24 hours after the beginning of your assigned lab.]

* see tar tutorial on page 2.

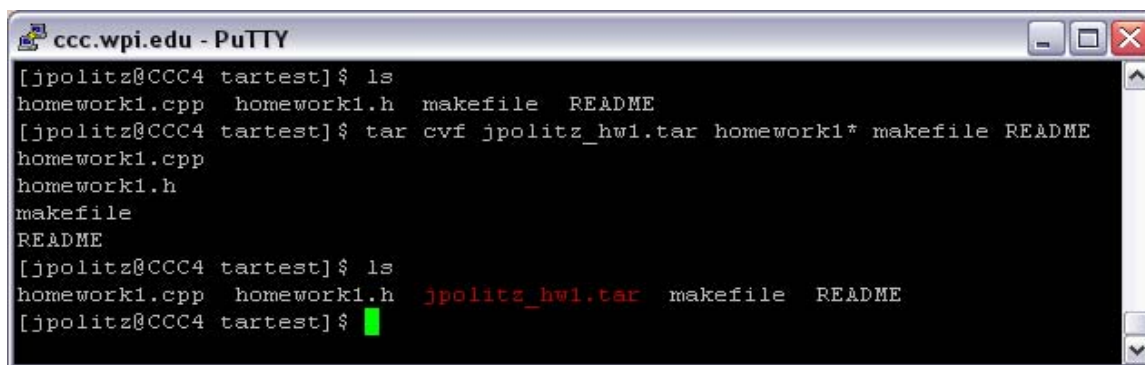
** see http://web.cs.wpi.edu/Help/turnin.html#students_help for instructions on how to use turnin.

Using the tar command in UNIX

For class assignments and labs, you must submit a *tar* archive file of your work. Place all the files that you wish to *tar* in the same directory. From that directory, type a statement of the form:

```
tar cvf archive_name file1 file2 ...
```

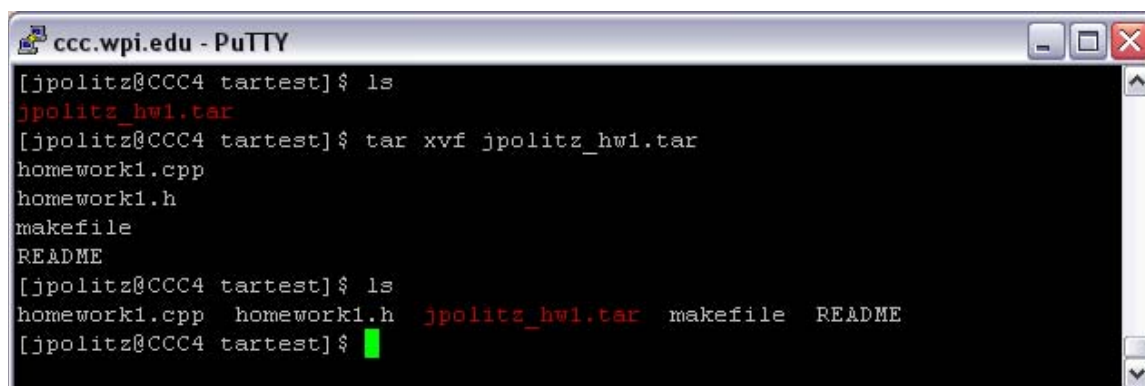
The “cvf” represents three possible options: **c** means “create”; **v** means “verbose” (filenames are echoed as they are added to the archive file); and **f** tells the *tar* command that the archive will be created from the files you list. The archive name will be the name of the tar file that is created, and filenames are files that will be archived. An example:



```
ccc.wpi.edu - PuTTY
[jpolitiz@CCC4 tartest]$ ls
homework1.cpp  homework1.h  makefile  README
[jpolitiz@CCC4 tartest]$ tar cvf jpolitiz_hw1.tar homework1* makefile README
homework1.cpp
homework1.h
makefile
README
[jpolitiz@CCC4 tartest]$ ls
homework1.cpp  homework1.h  jpolitiz_hw1.tar  makefile  README
[jpolitiz@CCC4 tartest]$
```

The file in red, “jpolitiz_hw1.tar” is the tar archive. This is the type of file you will submit to *turnin* for your assignments and labs.

To extract a tar file, using the option **x** instead of **c** will allow you to extract a given tar archive to the current directory. Prior to your final submission, you should do this to test if the creation of the tar file worked the way you expected. Remember, if you turn in a tarred file that has errors, it is possible that you will receive no credit for this assignment! An example:



```
ccc.wpi.edu - PuTTY
[jpolitiz@CCC4 tartest]$ ls
jpolitiz_hw1.tar
[jpolitiz@CCC4 tartest]$ tar xvf jpolitiz_hw1.tar
homework1.cpp
homework1.h
makefile
README
[jpolitiz@CCC4 tartest]$ ls
homework1.cpp  homework1.h  jpolitiz_hw1.tar  makefile  README
[jpolitiz@CCC4 tartest]$
```

This should be all you need to know about *tar* in this course. For more information on the use of other options, refer to the *tar* man page. (i.e., type “man tar” from the command line).