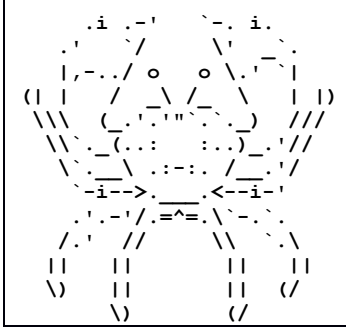


The University of New Mexico

CS 241—Data Organization

Lab 1: Introduction to C, ASCII ART & the Linux Command Line

Brief Introduction to Command Line Linux:

This lab is designed for you to have a chance to familiarize yourself with the computing environment we will be using this semester, and in a number of upper level CS classes.

Getting a CS account and Logging in to Linux from the pods:

Each student enrolled in CS-241 needs a CS Linux account: visit the CS support staff with your UNM ID card. Inform CS support that you are enrolled in CS-241 and request a CS account login.

Installed on the CS machines are the Linux operating system, Ubuntu, and a shared file server. When you login, the active directory will default to your *home directory* on the shared file server. Anything you save on this shared file server will be visible to you on future logins regardless of which CS machine you are using.

There are a few ways that you can log in to a Linux system. Some of the physical machines are located in the FEC student labs. This is a great place to do homework as you will meet other CS students and the department pays talented senior CS students to tutor and/or help CS students with 100 and 200 level CS classes.

The UNM IT Windows computers in the classrooms and pods all have installed an SSH (Secure Shell) program called Telnet. You will use Telnet to make a text only login connection (called an SSH login) to the CS Linux machines. CS supports three group server machines:

shuttle.cs.unm.edu
moons.cs.unm.edu
gigs.cs.unm.edu

MacOS ships with a pre-installed SSH client.
PuTTY, a free Windows SSH client can be downloaded from:
http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html.

Security Note: If you are going to install PuTTY, copy or click the link above rather than doing a Google search for "PuTTY". PuTTY is a perfectly safe program. However, since it has many users, some nefarious characters have downloaded PuTTY, mixed in their own pet malware, repackaged it and offer it for free on various websites for which a Google search will find and happily provide a link.

You can use Telnet to open a connection to one of these which will automatically attempt to open session for you on which machine in its child group currently has the least load. Sometimes you will not be able to connect to one of these or your will find the connection very laggy. When that happens, try another. You can also connect directly to any of the machines in the CS lab by entering their host name (printed on the top of the machine) followed by "**.cs.unm.edu**".

When your work is graded, it will be run on one of the CS Dell Precision T3610 machines (Intel Xeon CPU E5-1607 v2 @ 3.00GHz, 16 GB RAM) located in the ECE 332 student lab (**brown**, **setebos**, **sycorax**, **leda** or **metis**)

Basic Linux Commands:

Hopefully now you are logged in, and on your screen you have and empty window that has a text line in it that looks something like this:

username@computer:~$

This line is called the command prompt or simply prompt. In Linux, the "My Documents" of Windows®, is called your home directory. When you first start out, your home directory will contain a number of default files and folders. To see what is present in your home directory, enter the command "**ls**" (without the quotes) on the command prompt and hit return. The "**ls**" command has a number of options:

- **ls -F** appends a "*" to the end of every executable file, a "/" to the end of every folder, an "@" to end of every symbolic link, and a few other suffixes for other special file types.
- **ls -l** creates a more verbose listing of the files and folders in the directory that includes permissions, size, date modified, the user name of the owner, and some other information.
- **ls -a** includes *hidden* files: that is files with names whose first character is ".". These generally include various configurations. Some of these files are automatically executed during the login process. Some are executed when a particular application starts. One file of particular note is the **.bashrc** file. This file is a plane text, batch file that runs automatically run during login. Unless you know what you are doing, you should not change anything in this file; however, it is generally safe add net commands to the end of the file. For example, many people append the Linux command **"alias ls 'ls -F'"** to the end of their **.bashrc**. This creates an alias for ls so that whenever "**ls**" is entered as a command, Linux will replace it with "**ls -F**". The single quotes are needed because the string "**ls -F**" contains a space.
- **ls -laF** applies all three of the above effects.

Find out more about **ls** by reading the *manual pages*. The command "**man ls**" at the command prompt will display the manual pages (often called man pages).

Creating Directories:

Create a new folder (called directory in Linux) with the **mkdir** command. For example, if you want to create a folder called cs241 enter the command: **mkdir cs241**.

Navigating Directories:

Next up, we'll try to descend into the newly created directory. This can be done by using the **cd** (change directory) command: **cd cs241**.

Now that the active command location has entered the newly created directory, enter the **ls -aF** command, and you will see that the directory is empty except for two directories: "**./**" and "**../**". The first of these is a pointer to the current directory and the second is a pointer to the directory one level up. Thus, "**cd ..**" will return the active command location back to your home directory. Additionally, no matter where you are in a directory structure, you can always return to your home directory with the command "**cd ~**".

Editing text files with vim

Vim is a model text editor. Start vim with the command: "**vim helloWorld.c**".
Before you can start typing in vim, you must use the i command to enter *insert mode*.
See the class notes (http://www.cs.unm.edu/~joel/cs241/note/CS-241-Lecture-02-HelloWorld-InLinux.pdf) for a description of basic vim commands (such as *end insert mode*, *save file* and *exit vim*).

Copying, Renaming, Moving, and Removing Files:

The Linux commands to copy, move and delete (remove) a file are:

```
cp existingFileName NewfileName
mv existingFileName NewfileName
rm filename
```

For example:

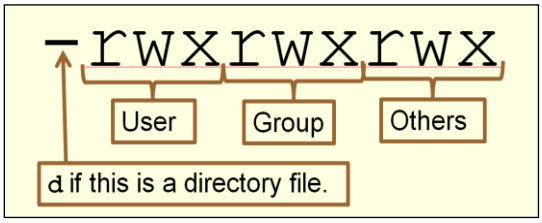
```
cp helloWorld.c myBackupCopy.c
```

will create a copy of **helloWorld.c** with the name **myBackupCopy.c**.

Setting File and Directory Permissions:

The **chmod** command (abbreviated from change mode) is a shell command in Linux environment used to change file system permissions (also called *file mode bits*) of files and directories.

The **ls -l** command will show a files permissions in for format:



The syntax of the **chmod** command is:

chmod <*who*><+|-><*permission*> *file*

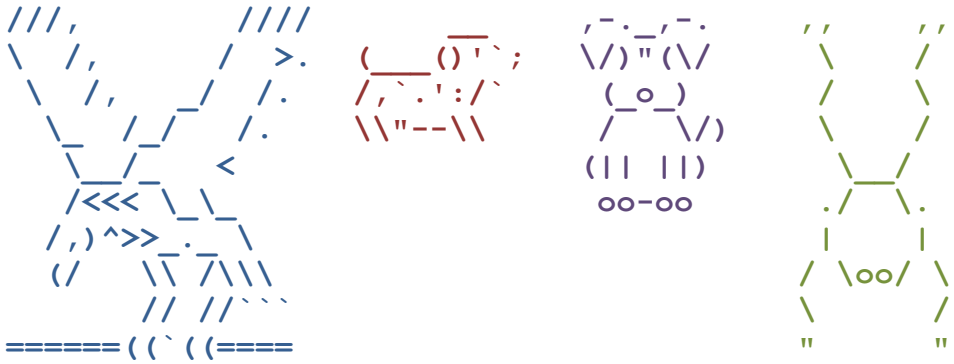
who	u (for user), g (for group), o (for other), or a (for all)
+	add permission
-	remove permission
permission	r (for read), w (for write), or e (for execute)

For example, in order to make all files in a directory unreadable by anyone other than you use the commands:

```
chmod o-r *
chmod g-r *
```

ASCII ART

ASCII art is a graphic design technique that consists of pictures pieced together from the 95 printable (from a total of 128) characters defined by the ASCII Standard from 1963 and ASCII compliant character sets. ASCII art can be created with any text editor, and is often used with free-form languages. Most examples of ASCII art require a fixed-width font (non-proportional fonts, as on a traditional typewriter) such as Courier for presentation. Examples:



Problem Specification:

Use the SSH client to connect to a CS machine with your CS account.

1) Create a new working directory called **cs241**, and within that directory create another directory called **lab-01**.

2) Use the Linux shell **chmod** command to ensure that only the owner has read, write and execute permissions for the two new directories.

3) Within this directory, you are to create a C program, stored in the file **asciiArt.c**.

4) Your **asciiArt.c** must use the **printf** function to display to the standard output stream an ASCII art image that fits within 60x24 columns and rows. It is ok to be smaller than that size. Your image must be a stylized version of your first or last name. Each "letter" must be at least 3x3 characters: **printf("Joel\n")** would NOT get credit as an ASCII art image of my name.

   The first five lines of your **asciiArt.c** must be of the form:

```
/************************************************/
/* YourFirstName YourLastName                   */
/* Date                                         */
/* CS-241 Section #                             */
/************************************************/
```

5) Use the Linux shell **chmod** command to ensure that only the owner has read, and write permissions for your **asciiArt.c**.

6) Compile the program with **/usr/bin/gcc**.

7) Compile and run your modified program with the output redirected from the shell to a text file called ***yourfirstname-yourlastname.txt***. This is done using a shell redirection "**>**" command:

```
./a.out > yourfirstname-yourlastname.txt
```

8) Use a text editor to edit ***yourfirstname-yourlastname.txt***. To the *beginning* of the file, add the following items:

   a. Your name
   b. Your major
   c. Your cs login name
   d. Why you enrolled in this class (this can span more than one line)

9) From the **lab-01** directory you created in step 1, your run the Linux command:

```
pwd >out1.txt
```

   This will create a file that shows the directory path you created. It must be the correct directory path on **cs.unm.edu** of your CS account home directory with the directory created in step 1.

10) From the directory you created in step 1, your Run the Linux command:

```
ls -a -l >out2.txt
```

This will create a file, **out2**, containing the current directory (.), the parent directory (..), your **asciiArt.c** file, the executable file **a.out**, and whatever other files you have in the directory. The **out2.txt** file you created must include the permissions, owner, group, size and date of each file. The permissions of the current directory and your c program must only be read/write for the owner.

Turning in Your Assignment:

You are now done with your assignment, and need to turn in the files **asciiArt.c** ***yourfirstname-yourlastname*****.txt**, **out1.txt** and **out2.txt** (not **a.out**). Since there is more than one file to turn in, please compress them into what is called an archive with the Linux command:

zip *my.zip file1* [*file2*] [*file3*] [...]

For example, the command:

zip *yourname***.zip *.c *.txt**

will create a zip archive that contains all files in the current directory matching either of the two wildcard expressions (all files ending with .c and all files ending with **.txt**).

By the way, it makes life much easier for the grader when you name your .zip archive with your name. In general, it is wise to make your grader's life easy.

Finally, you need to attach the zip archive into Blackboard Learn.

If you are in the CS lab seated at the CS machine, uploading to Blackboard Learn is easy: Open a browser, open Blackboard Learn, go to the assignment turn-in page, browse for the .zip archive, click "attach" and click "submit" (do not forget to click submit or your assignment will not be delivered to the grader).

If you are using a remote telnet session, then before you can use the browser to upload the .zip archive, you will need to copy the file from the remote CS machine to your local machine. Do this by using a SFTP (Secure File Transfer Protocol) client. The UNM lab machines all have an SFTP client installed (it is listed as FTP, but the client is actually secure). MacOS computers ship with a built-in SFTP client. A free Windows client is WinSCP (https://winscp.net/eng/download.php).

Grading Rubric (total of 20 points)

[1 point]: The program starts with the specified comments. Note: the number of '*****' can be any number less than or equal to 78 and large enough so that all the terminating '***/**' are aligned.

[2 points]: The four files are named as specified.

[2 points]: The program compiles without errors or warnings on ***x*.cs.unm.edu** using **/usr/bin/gcc** with no options.

[2 points]: All code in each block is indented exactly two spaces per block structure level as shown. *Spaces* must be used for indenting, *not tabs*. Note: in this simple program, most likely your only block will be the **main** function.

[1 point]: No characters except spaces are on the same line as any open, {, or close, }, curly brackets.

[5 points]: When compiled and run, the program outputs a beautifully stylized, and readable, ASCII art version of your name.

[3 points]: The directory structure shown in **out1.txt** is as specified.

[4 points]: The file permissions of **asciiArt.c** and its parent directory are shown in **out2.txt** and are as specified.