(COMP-1400) Lab Exercises #1

OBJECTIVE:

The first objective of this lab is to make students familiar with the Linux command-line environments and basic operations. Additionally, the students will write, compile and execute their first C program in this course.

To complete these lab activities, you first need to connect to the CS servers remotely. You can use the NoMachine software to get access to the Linux servers' desktop. You can download the NoMachine client software from "https://www.nomachine.com/." A tutorial is available online at https://help.cs.uwindsor.ca/captivate_packages/NoMachine/. More tutorials are available on the course website (Resource section). In addition, check Lab 0 for more details and tutorials.

If you use Mac computer or a computer with Linux operating system, alternatively you can use your own computer to do the lab activities, although working on CS server is recommended to make sure you do not lose your files and also using the most recent versions of the compiler/software tools.

Part A: Linux Exercises

Example: Login to the system, open the **Terminal** and type the following command:

- **echo** hello world

Comment: This command displays a line of text (e.g. hello world).

To read what an **echo** command does you can type:

- **man** echo

Comment: Using **man** command you can display user manual of Linux/Unix commands. Press q to exit the manual page and return to the command line.

To know more about this command type the following command:

- **man** man

To read user manuals of linux commands you can simply use:

- **man** < command name >

When we run `` `**man** echo` ``, we see the following:

NAME

***echo** -- write arguments to the standard output*

SYNOPSIS

***echo** [-n] [string ...]*

We can learn how to use the command using the synopsis. For example this synopsis uses [] to show optional arguments. This means echo can be ran without arguments like `` `**echo**` ``. Optionally, we can include a string (ex. `` `**echo** *"example string"*` ``) and the **-n** argument (ex. `` `**echo -n**` ``) which is described later in the manual:

-n   *Do not print the trailing newline character.*

Getting started: Save the commands used to solve the next 3 exercises in a file called "PartA.txt" using an editor (like gedit or nano). You can include the outputs below the command used.

Exercise 1: Display your name using an echo command.

Exercise 2: Study the following Unix/Linux utility commands using their **man** pages and <u>execute them</u> on the terminal. The manual is used to learn the commands, then you should try using the command in their most relevant form (only include how to use the command in PartA.txt, not the manual).

who	mkdir	cp
ls	rmdir	mv
cd	cat	rm
Date, time	pwd	chmod

Example:

1) Type the following command to print the current working directory (term "directory" in Linux/UNIX means exactly the same as term folder in Windows).
   - **pwd**

As a result of executing this command the system displays your current directory (folder).

2) Type **ls** command to list current directory.
   - **ls**

This command displays the contents of your current directory (folder).

Note: Some commands will require a file to move, display, delete, or copy. Use a text edit to make an example file.

Exercise 3: Accomplish the following tasks, and record your list of commands in PartA.txt for the sequence of tasks.

a. At the terminal prompt, create a directory for the semester in the base directory (ex. **winter**).
b. In a similar manner, create a **comp1400** directory in the newly created directory.
c. Change your working directory by going to the **/baseDir/season/comp1400** directory, example if we are in winter using the school server, the directory would look like **/home/username/winter/comp1400**. We can also use ~ to represent the baseDir as a shorthand, example **~/winter/comp1400.**
d. Check the name of your current working directory.
e. List all the files of the current directory, including hidden "dot" files (check manual of the command or help document at the end of the lab PDF).
f. Record the list of commands and outputs in PartA.txt

Note: What are "dot" files? A file prepended with a dot, for example ".example_file" are hidden from the interface and are not generally shown to GUI interfaces or normal usage of `ls`. Other dot files exist such as "." and ".." which are used to link folders. The "." file represents the current folder, generally used for paths like when you run files in Part B (./myProg), so you access the current folder for its contents. The ".." file represents the previous folder. If you wanted to change directories to the previous folder, you can try `cd ..` which will move you one folder back. Use `pwd` before and after to see how your path changes.

Part B: Basic C Programming

Exercise 1: Start a text editor, which could be **gedit** (or **vi** or **nano**) on a Linux machine (Notepad or notepad++ on PC).

Write the following simple C program.

```
#include <stdio.h>

int main(void) {
        printf("Welcome to C!\n");
        printf("************\n");
        printf("Good Bye!...\n");

}
```

a. Save the file in the format of a C source file as **firstPrg.c** in your working directory, i.e., **/home/yourUserName/fall/comp1400**.
b. As you are already in the working directory as completing UNIX Exercise, you can now compile the C program with the following command to produce an executable.

- **gcc -Wall firstPrg.c -o myProg**

c. Run the executable with the following command to produce the output (step b must not have errors, in other words have no output).

- **./myProg**

Exercise 2: After compiling and running this program, modify the code by replacing the asterisks (the second printf) with your own name. Then re-compile and run the program to see the changes, including printing your name on screen!

Lab Evaluation:
The lab will be graded in person by a GA or TA. Notify one when you are done and they will provide feedback and/or a grade. Show them PartA.txt and how to compile and run your program in Part B (exercise 2 is fine). A TA/GA might ask you to explain a command in part A or different parts of the lab.

You can submit the files on Brightspace as a backup, but all grades are given in person.

Total 10 marks.

COMMON COMMAND AND CONTROL SEQUENCES

The following commands can be entered at the UNIX system command prompt. Press the ⟨Return⟩ key after each command. Use lower case letters unless otherwise indicated.

Command	Description
WORKING WITH DIRECTORIES	
pwd	Gives you the name of your current working directory.
cd *dirname*	Changes your working directory to *dirname*. If you are not in *dirname*'s parent directory, you must use *dirname*'s full pathname.
cd ..	Changes your working directory to the parent of the directory that you are in when you issue this command. If you are at the root, you will remain there.
cd	Changes your working directory to your home directory.
ls	Lists the files in your working directory.
ls -l	Lists the files in your working directory, giving lengthier information about them — including file protection and access privileges.
ls -a	Lists the files in your working directory and shows your hidden "dot" files, as well, including your .login file.
ls -lt	Same as ls -l, but orders the files by time modified.
mkdir *dirname*	Creates a directory called *dirname*. Your working directory will be the parent of this new directory, unless you specify another pathname.
rmdir *dirname*	Deletes a directory called *dirname*, if the directory is completely empty. If you are not in *dirname*'s parent directory, you must use *dirname*'s full pathname.
WORKING WITH FILES	
pico *filename*	Uses the editor pico to open a file called *filename*. If a file of that name does not already exist, it starts a new one.
vi *filename*	Uses the editor vi to open a file called *filename*.
cp *file1 file2*	Makes a copy of a file called *file1* and names the copy *file2*. If there already is a file called *file2*, it will be replaced.
cp -i *file1 file2*	Makes a copy of *file1* and calls the copy *file2*. If there already is a file called *file2*, the system queries you before replacing it.
rm *filename*	Deletes a file called *filename*. If you are not in the same directory as *filename*, you must specify *filename*'s pathname.
rm -i *filename*	Deletes a file after confirmation. Type y to confirm, n to cancel.
more *filename*	Lists the contents of the file called *filename* on your screen. Press the spacebar to scroll forward in the file. Type b to scroll back one screenful. Type q to quit.
lpr -P *pr fname*	Sends a file called *fname* to a printer called *pr*.

ADDITIONAL COMMANDS	
history	Lists your previous commands. !n Executes command number n in the list that you receive as output of the history command.
man *command*	Obtains online information about the command of your choice. These are the UNIX system's online "man pages". Press the spacebar to scroll to the next screenful. Type b to scroll back a screenful. Type q to quit the man facility.
man -k *key*	Obtains online information when you do not know the command name. Replace *key* with a keyword for the function you wish to perform.
learn	Initiates online instruction in file manipulation, vi, C, and other topics.
passwd	Initiates the password-changing program. Respond, as prompted, with your old and new passwords.
pine	Starts the Pine mail program.
tin	Starts the network news reader tin. Use the up- and down-arrow keys to highlight a newsgroup (or message) of interest; press Enter key to select it. Type ? for help, and q to quit.
logout	Logs you off the UNIX system. If you have a "stopped process", the system will prompt you. Enter fg to resume the process or logout a second time to abort the process and log off the system.

KEYS AND TERMINAL EMULATION	
In the following, a caret (^) indicates the "control" key (often marked CTRL or CTL). Hold the "control" key down while typing the letter shown next to it.	
Backspace	To backspace, use the backspace key (often marked BKSP) or, if that does not work, the delete key (often marked DEL).
^c	Aborts the current process — e.g., a long screen listing.
^s	Temporarily suspends a listing until you type ^q.
^q	Resumes a listing after it has been suspended with ^s.

MORE CONTROL KEY SEQUENCES (ADVANCED USAGE)	
^u	Erases the whole line of characters you have typed thus far at the command line.
^d	This is the end-of-file indication.
^z	Suspends the execution of a process/"job". You may then have the process continue in the background (type bg immediately after). Type fg to resume the job or bring it to the foreground. To kill a suspended job, type jobs, then kill %n, where n is the number of the suspended job. (Note: use of this suspend character should be avoided until you have some familiarity with the system.)
^l	Redraws the screen.