Lab 1: Introduction to Linux

UWYO COSC 2030

Introduction:

Welcome to Data Structures! As you might expect, this class is primarily about data structures and the practical application thereof, but over the semester you will also learn about some related tools that will be useful to you in this class and in others after it. The primary function of the labs is to give you the time to learn these tools, so it's really in your best interest to show up and at least try to complete the lab in the time allotted. Labs are not designed to be very difficult, except for the challenge labs (because... challenge? you get it), which will occur in the week before each exam and (upon completion) will award you up to 5 extra credit points for the exams. This will almost certainly be the only extra credit opportunity you will have in this class, so do with that information what you will.

General expectations for all labs:

While collaboration is allowed and generally encouraged, you are expected to do and submit your own work. Remember, the same person grades all of the labs, and he (I) will notice if your code is too similar to somebody else's. Yes, even if you change your variable names.

When you submit, include a README.md file with your name and your lab section. You will be docked points if you forget, so don't.

Labs assigned on a Tuesday will be due the following Sunday at 11:59 pm. You will be docked points if you are late, and if you are more than 3 days late without clearing it with me first I will not accept your lab.

Brief note about challenge labs:

As stated above, challenge labs will happen in the week before each exam. They will be difficult, but not impossible. You will be given access to the lab instructions at the top of the lab section, and you will have until the end of the section to submit. Whatever you can accomplish in that time will be what you can submit, so bring your A-game.

Now, the actual lab:

Linux can be a bit daunting if you're used to having a GUI, but I promise it starts to make sense very quickly. You can use the following commands to help you get started:

ssh- secure shell. This is used to connect to another computer.

ls- list. This lists the documents and folders in your current directory.

pwd - print working directory. This prints the path to your current directory.

cd - change Directory. Used to navigate the file system. For example, cd <folder> or cd ..

g++ - the gnu C++ compiler. g++ example.cpp

touch- makes a file with the supplied name. For example, touch example.cpp

nano - opens the text editor for use. For example, nano example.cpp

vi- another text editor, works similarly to nano.

./ - running your program. We'll come back to this.

mkdir- Make directory. For example, mkdir folderName

rm- remove a file. For example, rm fileName

Example file creation/execution:

touch example.cpp (This step can be skipped though)

nano example.cpp

g++ example.cpp -o exampleOut

./exampleOut

Lab 1 (for real):

Step 0. Open the terminal you set up in lab 0

Step 1. Connect to the department Linux machines using ssh, once you have logged in use passwd to change your password. The initial password will be: changeme

Step 2. Use ls and pwd commands to see the results

Step 3. Make a folder called 2030_Labs to make a folder for these labs

Step 4. Move your current working directory into your new 2030_Labs folder, travel back up to your home folder, and return to the 2030_Labs folder

Step 5. Make a new .cpp file named lab1

Step 6. In nano/vi write Hello World

Step 7. Save and exit your text editor

Step 8. Compile your program, set the output to be labOne

Step 9. Run your program, address errors as needed

Submission:

There is nothing to submit this week.