CMSC 201 Fall 2015
Lab 01 – Hello World

Assignment: Lab 01 – Hello World
Due Date: Thursday, September 3rd, 2015 by 8:59:59 PM
Value: 1% of final grade

At UMBC, our General Lab (GL) system is designed to grant students the privileges of running a website, storing files, and even developing applications. GL itself is a Linux-based system that we will interact with using a command line interface (CLI). All major operating systems (Windows, Mac, and Linux) can log into GL.

In order to log into GL, you will need to have a Secure Shell (SSH) client. SSH is a network protocol that connects a client computer to a server securely, without the risk of exposing your password or any other sensitive data to anybody watching the network traffic. It is an alternative to Telnet, and is a leading standard in remote login. So, what does all of that mean? Simply put, SSH allows you to use your home computer to connect to a server (ie. gl.umbc.edu) and issue commands as if you were sitting at that computer. So, once you SSH into GL, you can do everything you would be able to do from a computer on-campus. This client is built into Mac and Linux. For Windows, we recommend downloading the open-source client PuTTY.

In Lab 1, we will be logging onto GL and setting up folders for 201 in your home directory. We'll also create a simple python program, and turn it in using the `submit` command. Finally, you'll send your TA an email with a bit of information about you, so they can get to know you better.

Part 1: Logging onto GL

For Windows Users Only

<u>Step 1A:</u>

Watch this video (https://www.youtube.com/watch?v=uE3VpOQi_GQ) , and download and install the PuTTY program. PuTTY will allow you to log into GL from your machine.

<u>Step 1B:</u>

Watch this video (https://www.youtube.com/watch?v=DH3of3tuBxA) and follow its instructions to log onto GL using PuTTY.

For Mac Users Only

<u>Step 1:</u>

Watch this video (https://www.youtube.com/watch?v=sHltxe2bLVI) and follow its instructions to log onto GL using the Terminal application already installed on your machine.

Part 2: GL and your home directory

In this part of the lab, we will log onto GL and create some folders in your home directory. These folders will be where you store your labs and assignments for 201 this semester.

If you would like a video explanation of how to navigate and create folders in GL, you can watch this (https://www.youtube.com/watch?v=_Jhtcs70G1U). Please note that this video does not show the correct folders to create.

Since GL is a CLI we can't navigate it using our mouse to click on folders. Instead, we must type in commands to tell it what we want to do. The solid green rectangle (or something similar) is where our cursor is. After typing in a command, you must hit enter for the system to know you are done.

Step 2:

After logging onto GL in Part 1, you are in your home directory (remember, directory is just another name for folder). We will first make a 201 directory in your home directory. The command we will use is `mkdir 201`. Remember that Linux commands are case sensitive. Type `mkdir 201` now, then hit enter.

```
linux3[1]% mkdir 201
linux3[2]% █
```

If you make a mistake in naming your folder, you have two options. You can either remove that directory or you can rename the directory. To remove the directory type `rm -r *mistakename` and hit enter. Be very careful that you do not remove the wrong directory. There is no recycle bin or "undo" button. The safer option would be to rename the directory. To rename the directory, you use the "move" command `mv`. Type `mv *mistakename lab1`.

```
linux3[2]% mv test123 lab1
linux3[3]% █
```

Step 3:

We now want to change our current directory to this new 201 directory. (In other words, we are going "inside" the 201 folder.) The command is **cd 201**, which means "change directory to the 201 folder." Type **cd 201** now, then hit enter.

```
linux3[2]% cd 201
linux3[3]%
```

Step 4:

In order to see the contents (like files and subdirectories) of a directory, we need to use the **ls**, or "list," command. Since the 201 directory was just created, it should be empty. You can check this by typing **ls**, then hitting enter.

```
linux3[3]% ls
linux3[4]%
```

Step 5:

We want to create two directories inside the 201 directory: one called Homeworks, and another called Labs. For this step, we again use the **mkdir** command. First type **mkdir Homeworks**, then hit enter. On the next line, type **mkdir Labs**, and hit enter again. Finally, type **ls** to verify that you now have two directories named Homeworks and Labs in your 201 folder.

```
linux3[4]% mkdir Homeworks
linux3[5]% mkdir Labs
linux3[6]% ls
Homeworks   Labs
linux3[7]%
```

Step 6:

We will next create a hw1 directory inside the Homeworks directory. When you work on your first homework, you will save your hw1.txt file in this directory. Right now, we just want to create the hw1 directory.

First, we need to change our current directory to be inside the Homeworks folder. To do this, type **cd Homeworks** and then hit enter. Next, to create the hw1 folder, type **mkdir hw1**. You can then type **ls** to check that the hw1 directory has been created.

```
linux3[7]% cd Homeworks
linux3[8]% mkdir hw1
linux3[9]% ls
hw1
linux3[10]%
```

At this point in the lab, your GL account has directories that look like this:
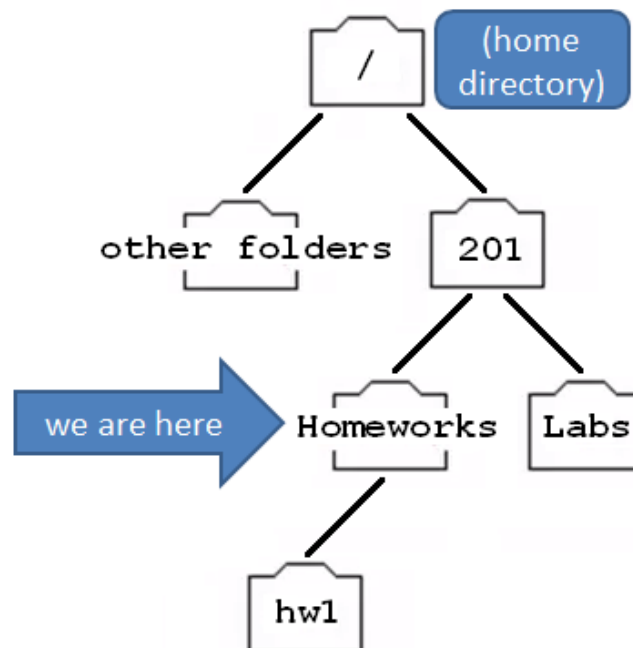
Step 7:

Now we want to create a lab1 subdirectory inside Labs. Before we can do this, we need to move to the Labs directory. First, we go up one "level," back to the 201 directory. To do this, we type `cd ..` ("cd" followed by a space, then two periods). The two periods (`..`) represents the parent directory. The parent directory of Homeworks is 201, so we are now back in the 201 folder.

```
linux3[10]% cd ..
linux3[11]%
```

Step 8:

Let's just double check that we are actually back in the 201 directory. We do this by checking our "present working directory," using the `pwd` command. Type `pwd` and hit enter. It will show you the full path, including "umbc.edu" and your username. If we look at the end, we can see that we are in the 201 directory, just like we wanted to be.

We are in the 201 directory, right underneath the home directory.

```
linux3[11]% pwd
/afs/umbc.edu/users/k/k/k38/home/201
linux3[12]%
```

This part of the path will depend on your username.

Step 9:

Now we can create the lab1 folder inside the Labs directory. First, we go into the Labs directory by typing **cd Labs**. Next, we create the lab1 directory by typing **mkdir lab1**. We can verify it was created by typing **ls** to list the contents of the Labs directory. Once it's created, we can move into the new directory by typing **cd lab1**. We can verify that lab1 is now our current directory by typing **pwd**.

```
linux3[12]% cd Labs
linux3[13]% mkdir lab1
linux3[14]% ls
lab1
linux3[15]% cd lab1
linux3[16]% pwd
/afs/umbc.edu/users/k/k/k38/home/201/Labs/lab1
linux3[17]%
```

We are in the lab1 directory, inside the Labs directory for 201.

This part of the path will depend on your username.

Part 3: Creating your first Python program

We are now ready to create our first Python program. Using the emacs editor, we will create a "Hello World" program, the traditional first coding project. You can optionally watch a video demonstration of this part of the lab here: (https://www.youtube.com/watch?v=oYIGpmUMZ8c).
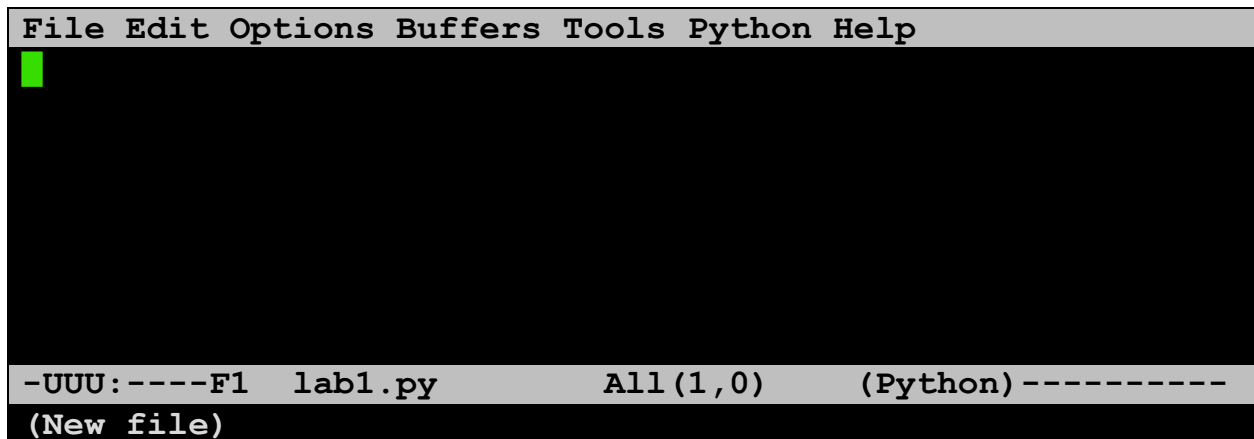
Make sure that when you complete this part of the lab, your current working directory is lab1!

Step 10:

Before we can begin, we need to start up the emacs editor. By giving the command **emacs lab1.py**, we are opening the file lab1.py in the emacs editor.

```
linux3[17]% emacs lab1.py
```

The emacs editor will replace your entire window of text, and should look similar to the example below. Since lab1.py does not already exist, we are given a blank file to work with.

```
File Edit Options Buffers Tools Python Help




                                                                        
                                                                        
                                                                        
                                                                        
                                                                        
-UUU:----F1   lab1.py         All(1,0)     (Python)----------
(New file)
```

Step 11:

Let's code our first program! This program will print out a short greeting when run. Type in the line of code shown below, but where the code says **FULL NAME HERE**, replace it with your full name.

```
File Edit Options Buffers Tools Python Help
print("Hello World! I am FULL NAME HERE!")

 -UUU:**--F1   lab1.py          All(1,42)    (Python)----------
```

It's important that you type in the code exactly as shown. Every character is important, and even small changes can keep the program from working. The only change should be replacing **FULL NAME HERE** with your full name.

Step 12:

Once you are done typing your code into emacs (and have replaced **FULL NAME HERE** with your name), you need to save your work and exit. First, save it by pressing **CTRL+X** and **CTRL+S**. (Press the CTRL and X keys at the same time, then the CTRL and S keys at the same time.)

If it worked, you should see a confirmation message at the bottom of your screen, similar to the one below (but with your username, of course).

```
 -UUU:----F1   lab1.py          All(1,42)    (Python)----------
Wrote /afs/umbc.edu/users/k/k/k38/home/201/Labs/lab1.py
```

Now that you've saved your code, you can exit by pressing **CTRL+X** and **CTRL+C**. You will exit out of emacs, and be back to the command line.

Part 4: Running your first Python program

You have the option to watch a video demonstration of this part of the lab here: (https://youtu.be/oYIGpmUMZ8c?t=2m53s).

Step 13:

We will be using Python 3 for all of your assignments. However, GL runs Python 2 by default, so we need to tell it we want to use Python 3 instead. To do this, type in the command **`/usr/bin/scl enable python33 bash`** and then hit enter.

```
linux3[17]% /usr/bin/scl enable python33 bash
bash-4.1$ 
```

You'll know it worked because the prompt changes. You must enable Python 3 every time you want to run a Python program! If your prompt starts with "linux" as it does in the first line above, you are not running Python 3.

Step 14:

To run your code, type in **`python lab1.py`** and hit enter. Your program should run as seen below, but with your name instead of **`FULL NAME HERE`**.

```
bash-4.1$ python lab1.py
Hello World! I am FULL NAME HERE!
bash-4.1$ 
```

If your program does not run, but instead prints out error messages, you have at least one syntax error. Just open the file up in emacs again (you can type **`emacs lab1.py`** after the **`bash-4.1$`** prompt) and take a close look at your code. Make sure it matches the code from Step 11.

For example, here's the error if the quotes are outside the parentheses:

```
bash-4.1$ python lab1.py
  File "lab1.py", line 1
    print"(Hello World! I am FULL NAME HERE!)"
                                               ^
SyntaxError: invalid syntax
bash-4.1$ 
```

Part 4: Submitting your first Python program

Step 15:
Watch this video (https://www.youtube.com/watch?v=r0otsJZ1ry0) for an explanation of how GL works. Follow the instructions in the video to submit your lab1.py file.

Step 16:
After submitting the file, we can verify that it went through by using the **submitls** command. The command needs to know what class and what assignment – these should be the same as for the **submit** command you just ran: cs201 for the class, and LAB1 for the assignment.

```
bash-4.1$ submitls cs201 LAB1
total 21
drwx------   2 mneary1 rpc  2048 Aug 29 15:27 .
drwx------ 507 mneary1 rpc 18432 Aug 27 15:01 ..
-rw-r--r--   1 k38     rpc    22 Aug 29 15:27 lab1.py
bash-4.1$
```

On the far right of the last line, we can see that the lab1.py file was successfully submitted. The other two files are the submit directory ("**.**" or a single period) and its parent directory ("**..**" or two periods).

Step 17:
To exit back to the linux prompt, type **exit** and hit enter. (You will need to re-enable Python 3 again if you want to run another Python program.) To exit from GL, type the same **exit** command and hit enter.

```
bash-4.1$ exit
exit
linux3[18]% exit
```

This will log you out of GL and close the terminal window.

Part 5: Emailing your TA
Step 18:
Watch this video (https://www.youtube.com/watch?v=ib47h0Dffbo) and follow the instructions to send your TA an email that includes some information about yourself.

You can find out who your TA is under the "Lab Sections" page on Blackboard. Match the section number from your CMSC 201 discussion section (you can find it on your schedule) to the section number in the table. Double check you have the right one by making sure the time also matches the discussion in your schedule!

Part 6: Wrapping Up
Step 19:
Watch this video (https://www.youtube.com/watch?v=kMffmS8Rf1Q) to learn how labs are graded.

Step 20:
Watch this video (https://www.youtube.com/watch?v=ci42iOOtTRI) for an explanation of the "life cycle" of remote programming on GL.

Step 21:
Get started on your Homework 1!

Special thanks to James Gerity who wrote a GL primer on his website:
http://userpages.umbc.edu/~jg5/logon-howto.html