CMSC 201

Homework 0 – Submitting on the GL Servers

Assignment: Homework 0 – Submitting on the GL Servers
Due Date: Wednesday, September 13th, 2023 by 11:59:59 PM
Value: 10 points

Homework 0 is meant to walk you through all of the details of how to submit your CMSC 201 files and check that they were submitted correctly.

If you run into any errors that you don't understand, email your TA for help, or visit the office hours of any TA or instructor.

At UMBC, the 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 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 (*i.e.*, **`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 this tutorial, 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.

Step 1: Getting the software needed

For Windows Users Only
Note: this section applies to users running on versions of Windows earlier than Windows 10. If you are running Windows 10 or 11, simply open up a Powershell window and proceed to step 2.

Go to: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html and scroll down to "**Alternative binary files"**. Under "**putty.exe (the SSH and Telnet client itself)"**, click on the "x86 **64-bit: putty.exe"** link to download the "putty.exe" file to your computer. Make sure to choose a location that you will remember.

(Note that it is unlikely, but possible, that you have a 32-bit machine. In that case, you'll need to click on the **32-bit: putty.exe** link. So, if the 64-bit install doesn't work, try the 32-bit.)

Navigate to the location where you downloaded "putty.exe". Double click the file to start the PuTTY program. It would be a good idea to create a desktop shortcut for PuTTY!

After you have opened PuTTY, type gl.umbc.edu into the Hostname text field at the top

For Mac Users Only
Watch this video (https://www.youtube.com/watch?v=UTU-CTH_xLw) and follow its instructions to log on to GL using the Terminal application already installed on your machine.

It is recommended that you do **not** resize the Terminal window horizontally, as the wider window often leads to very weird behavior with how the text is displayed, as seen in the red text in the screenshot below. (You may resize the Terminal window to be taller or shorter without having this effect, but do not make it wider or fullscreen.)

```
                place += 1
                password = input("Please enter a password: ")

    # if none of these if conditionals were true, then passwordValid will stillb    u       and          ft
he
        print("Thanks for picking the super-secure password", password)

main()
```

Step 2: GL and your home directory

Summary

Outline

Turn on screen reader support
To enable screen reader support, press Ctrl+Alt+Z To learn about keyboard shortcuts, press Ctrl+slash
GL Home Directory Layout

In this part of the assignment, 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.

Sit back and watch this first!! Then continue.

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

After logging onto GL in Step 1, you are in your home directory (remember, directory is just another name for folder). We will first make a cmsc201 directory in your home directory. The command we will use to do this is **`mkdir`**, which stands for "make directory." We also have to tell the system what the name of the directory will be. In this case, the name is "cmsc201", so type **`mkdir cmsc201`** now, then hit enter. Remember that Linux commands are case sensitive!

```
linux3[1]% mkdir cmsc201
linux3[2]% cd cmsc201
```

We want to create two directories inside the cmsc201 directory: one called Homeworks, and another called Labs. For this step, we again use the **`mkdir`** command. First type **`mkdir Homeworks`**, then hit enter. Once you've done that, type **`mkdir Labs`**, and hit enter again. Finally, you can 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 3 Creating Your Homeworks and hw0 directories

Follow the instructions in the day 1 lecture slides on blackboard. You are now in your **`home`** directory. You should have created the **`cmsc201`** directory,

and the **Homeworks** and **Labs** directories inside that. The first thing we want to do is go into the **cmsc201** directory, and then into the **Homeworks** directory.

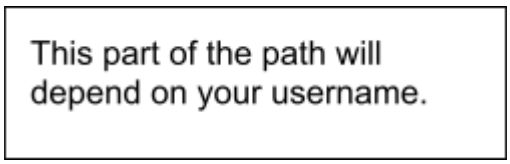
```
linux3[1]% cd cmsc201
linux3[2]% cd Homeworks
linux3[3]%
```

We need to create the directory to hold your Homework 0 file. We'll call the directory **hw0**, and then we'll go into it.

```
linux3[3]% mkdir hw0
linux3[4]% cd hw0
linux3[5]%
```

To double-check that you are in the right directory, type the "**pwd**" command to print the working directory (where you "are"). You should be in **home**, then **201**, then **Homeworks**, then **hw0**.

```
linux3[5]% pwd
/afs/umbc.edu/users/a/b/ab38/home/cmsc201/Homeworks/hw0
linux3[6]%
```



Now that you're in the correct directory, it's time to create a simple file. To create a new file, we use the **`touch`** command; type **`touch hw0.py`**, then hit enter. To open the file for editing, open it using emacs.

You do not need to touch the file if you are just going to open it with emacs in the next line, but if you want to create the file and not open it immediately, then touch will do that for you.

```
linux3[6]% touch hw0.py
linux3[7]% emacs hw0.py
linux3[8]%
```

Inside this file, we will want to start off with a "header comment" – we'll learn more about these later, but for now, you can use the following text, making sure to put in the date, your name, lab section, and username. In future assignments, you will have to write your own description.

```
"""
File:         hw0.py
Author:       Your Name
Date:         Today's date
Section:      Your Lab Section
E-mail:       your.username@umbc.edu
Description:  This file is to help students practice
              using the submission system on the GL
              servers for CMSC 201.
"""
```

You should have a "header comment" similar to this at the top of every file you submit for CMSC 201 this semester. Please make sure that you change the blue text, to your own information. Submitting this generic header without your information just plain looks bad.

Below the header comment, you'll write a single line of Python code.

```
print("This is going to be a great semester!")
```

Once you're done, you'll save 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.) Then exit emacs by pressing **CTRL+X** and **CTRL+C**.

After you create or update one of your Python files, you should always test that it works correctly. Making one small typo can cause a file to not run the way you want it to, and you wouldn't want to submit a "broken" file.

```
linux3[8]% python3 hw0.py
This is going to be a great semester!
```

If your program doesn't work at this point, open your **hw0.py** file again (using emacs), and look closely to see if you made any typos. Fix them, save your file, exit emacs, and try again.

Step 4: Submitting your hw0.py file

THE STEPS BELOW CAN ONLY BE DONE IF THE COORDINATOR HAS SET UP THE SUBMIT DIRECTORY. PLEASE WAIT UNTIL IT HAS!!
(there will be an announcement once the submission directory has been made)

Once you are done with your **`hw0.py`** file, you'll need to submit it to the **`cmsc201`** submission directory, and into the correct assignment directory. In this case, the assignment is **`HW0`**, so that's the one you'll submit to.

```
linux3[8]% submit cmsc201 HW0 hw0.py
Submitting hw0.py...OK
linux3[8]%
```

If you've already submitted a file, the system may prompt you to confirm that you want to overwrite your previous submission. If you want to overwrite your previous file with the one you've been working on, hit '**`y`**' for yes.

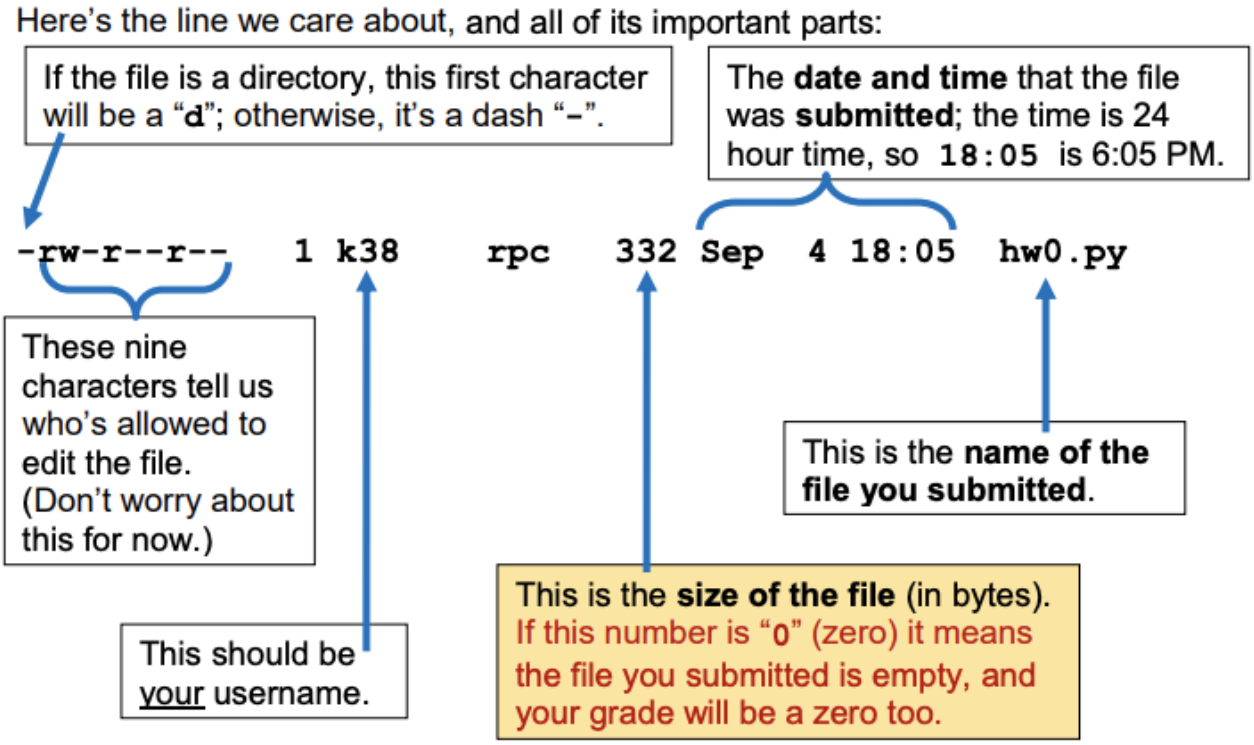
```
linux3[8]% submit cmsc201 HW0 hw0.py
It seems you have already submitted a file named hw0.py.
Do you wish to overwrite? (y/n):
y
Submitting hw0.py...OK
linux3[8]%
```

Step 5: Checking your submission

After you submit, you'll need to check that your submission went through, and that you submitted the correct file. We'll start by asking the **`submit`** system to list all the contents of our submission directory, by using **`submitls`**.

```
linux3[8]% submitls cmsc201 HW0
total 21
drwx------   2 emcgov1 rpc  2048 Sep  4 18:09 .
drwx------ 528 emcgov1 rpc 18432 Aug 31 16:54 ..
-rw-r--r--   1 k38     rpc   332 Sep  4 18:05 hw0.py
linux3[8]%
```

This gives us a lot of information, so let's break it down. We can ignore the first three lines: the "**`total`**", and the two folders called "**`.`**" and "**`..`**" (one and two periods, respectively). What we care about are any remaining lines of text. In this case, because we submitted only one file, there's only one line containing important information; later homeworks may have multiple files, so there will be multiple lines of important information, each about a different file.



Now let's take a look at the file you've been working on in your current directory, and see if it's the same size. You'll want to ask the system to list the contents, but we're also going to use "**`-l`**" (dash, and then a lowercase "L") to get a bit more information.

```
linux3[8]% ls -l
total 2
-rw-r--r-- 1 k38 rpc 332 Sep  4 16:54 hw0.py
-rw-r--r-- 1 k38 rpc 115 Sep  4 14:15 hw0.py~
linux3[8]%
```

You should see something like this. If your file was created all in one go, you probably won't have that second file with the "**`~`**" (tilde) at the end. (If you want to know what this file is, you can read about it on the last page.) For now, let's focus on the "**`hw0.py`**" line alone. What we want to do is check that the size of the **`hw0.py`** file in our directory is the same as the one in the submit directory. In this case, both files are **`332`** bytes, so we know that we have an up-to-date file submitted.

If your submit directory contained an old version, the sizes of the files would be different, and you would want to use the **`submit`** command again. The dates and times being different are okay, but the size of the file should be the same. (The file in your directory will have the date and time of your last edit; the file in the submit directory has the date and time it was submitted.)

If your file sizes match, then you are done submitting your assignment!

You can use these instructions for any assignment this semester – just replace "**`hw0.py`**" with the correct file name (or file names) and replace the "**`HW0`**" submission folder name with the assignment (*e.g.*, **`HW2`**, **`PROJ1`**).

If the file you submitted is zero bytes, keep reading.

Troubleshooting: File size of 0 bytes

If the file in your submission folder is showing up as being 0 bytes, the most likely cause is that you accidentally submitted a *directory*, not a file. The submit system doesn't allow you to submit an entire directory, and instead will just create an empty file with the same name.

The empty file problem most often happens when students accidentally submit their "**`hw0`**" directory, rather than the "**`hw0.py`**" file. Some students may even accidentally create a "**`hw0.py`**" *directory*, and then have a "**`hw0.py`**" *file* inside – this gets very confusing! To check if you've submitted a directory, use the "**`ls -l`**" command to get information about the files you're working on, and take a close look at the very first character.

```
linux3[8]% ls -l
total 4
drwxr-xr-x 2 k38 rpc 2048 Sep  4 18:51 Homework0.py
linux3[8]%
```

Do you see the problem with "**`Homework0.py`**" above? Its first character is a "**`d`**" instead of a "**`-`**", which means it's actually a *directory*. If we submitted "**`Homework0.py`**", the file in the submission directory will be empty (0 bytes):

```
linux3[8]% submit cmsc201 HW0 Homework0.py
Submitting Homework0.py...OK
linux3[8]% submitls cmsc201 HW0
total 21
drwx------   2 emcgov1 rpc  2048 Sep  4 18:53 .
drwx------ 528 emcgov1 rpc 18432 Aug 31 16:54 ..
-rwxr-xr-x   1 k38     rpc     0 Sep  4 18:53 Homework0.py
linux3[8]%
```

If you've submitted a directory rather than a file, don't worry about deleting it from the submission system (your TA will ignore extra empty files). Instead, you need to find your actual homework file and submit it. It should be "somewhere around here" – depending on how you organize your folders, it

may be inside the directory you tried to submit, in the same folder as it, or somewhere else entirely.

First try looking inside that directory, with "**`cd hw0`**" or "**`cd Homework0.py`**" or whatever your directory is called, followed by "**`ls -l`**". If the file is inside that directory, go ahead and follow the instructions from Step 4 and Step 5 again, and you should have a correctly submitted file.

If it's not in that directory, try checking other places in your GL account. If you ever get "lost," you can type "**`cd`**" (followed by no other text) and hit enter, and you will be taken back to your "**`home`**" directory to start over.

If you run into a problem like this, and you can't figure out how to solve it, go to the TA or instructor office hours for help.

If no one has office hours, you can ask a fellow CMSC 201 student for help! This falls under "Getting or receiving help with using GL or the UMBC Linux system," which is allowed regardless of the collaboration policy for the assignment.
(If the assignment is individual work only, just make sure you don't show them any of your code.)

Troubleshooting: Error Messages

Behind the scenes, submit is simply copying a file from your directory (the "source" file) into the submission directory we've set up (the "destination" file). Errors made when typing in the command can cause a number of different problems; here are four common ones, what they mean, and how to fix them.

Wrong class:

```
linux3[8]% submit CMSC201 HW0 hw0.py
Cannot find class CMSC201 in the system submit.class file
Perhaps you selected the wrong class?
linux3[8]%
```

The course you should be submitting to is "cmsc201" in all lowercase (remember, all Linux commands are case sensitive). If you try to submit to a course that does not exist, you will see that the system "cannot find [the] class" in the system. Make sure you've typed in the course name correctly.

Wrong assignment:

```
linux3[8]% submit cmsc201 HW1 hw0.py
This project is not defined in the instructor's submitrc.
The instructor doesn't seem to have a directory set up for
this project
(perhaps he/she didn't run submitinit?)
debug:
linux3[8]%
```

If, on the other hand, you type in an incorrect assignment name (or an assignment that has not been set up yet for submit), you will receive an error that states that the "project is not defined in the instructor's submitrc." Make sure you've typed in the correct assignment name.

Cannot open source file:

```
linux3[8]% submit cmsc201 HW0 hw0.yp
Submitting hw0.yp...cannot open source file hw0.yp
linux3[8]%
```

If you make an error when typing the name of the file you're trying to submit, you'll get an error that submit "cannot open [the] source file." Make sure you've typed in the filename(s) correctly.

Another possible cause for this error is if you're not currently in the directory containing the file you're trying to submit. For example, if you're in the "Homeworks" directory, and are trying to submit a file contained within the "hw0" directory, you will receive the same error. Type "ls" to double-check that you are in the same directory as the file you're trying to submit.

Cannot open destination file:

```
linux3[8]% submit cmsc201 HW0 hw0.py
Submitting hw0.py...copy: cannot open dest file
/afs/umbc.edu/users/e/m/emcgov1/pub/cmsc201/fa17/Submissio
ns/HW0/k38/hw0.py
linux3[8]%
```

Finally, if you attempt to submit an assignment after the due date has passed, you will receive an error that submit "cannot open [the destination] file," followed by the path to the file it's attempting to copy into. (Your error message would have your username instead of "k38" in the path.)

If you receive this error, that means the due date for the assignment has passed, and the system is simply stopping you from submitting to a directory you no longer have access to. Make sure that you submit your assignments early (and often) in order to avoid this issue.

Side Note: The "~" file

Remember that "~" file we saw when we typed "**ls -l**" in Step 5?

```
linux3[8]% ls -l
total 2
-rw-r--r-- 1 k38 rpc 332 Sep  4 16:54 hw0.py
-rw-r--r-- 1 k38 rpc 115 Sep  4 14:15 hw0.py~
linux3[8]%
```

We call that file the "tilde file" (pronounced "til-da") in computer science. If you take a look at the details, you'll notice two things about **hw0.py~** that are different from the "regular" **hw0.py** file.

1. It's an older file: it was last edited at "**14:15**" (2:15 PM), and the non-tilde file was last edited at "**16:54**" (4:54 PM).
2. It's a different size than the regular file. In this case, it's smaller (**115** bytes compared to **332** bytes), but that's not always the case.

The tilde file is a backup of what your work looked like the previous time you closed emacs. The emacs program creates it as a favor to you, to allow you to see a slightly older version of the file if you want to. You can safely ignore these files – they won't hurt anything.

Side Note: The "#" file

You might also see files that have "**#**" at their beginning and end.

```
linux3[8]% ls -l
total 2
-rw-r--r-- 1 k38 rpc 334 Sep  4 17:22 #hw0.py#
-rw-r--r-- 1 k38 rpc 332 Sep  4 16:54 hw0.py
linux3[8]%
```

These are what emacs creates if your file isn't saved properly (if your GL session crashes, if you choose not to save changes made, etc.). You can safely ignore these files as well. If you do need to recover the file (because

GL crashed or you accidentally closed the window), come see a TA or instructor in office hours, and we will walk you through it.