Assignment 1
CSc 210 Fall 2017
Due September 7th, 8:00pm MST

Introduction

In class we have been discussing using a command line to accomplish tasks on your computer. We have also discussed using git to provide revision control for a project. On this assignment, you will be getting comfortable with the github classroom interface for turning in projects as well as exercising some command line based skills.

The most difficult part of this project will probably be setting up git. Here is a tutorial provided by the creators of git that may be useful throughout the assignment: https://git-scm.com/docs/gittutorial

Your assignment will be turned in via github classroom. You will receive your own private repository (outlined below) and submit files into there. Your submission will be considered as the files present in that repository when the due date is reached.

Specification Part I: Setting up your repo

One of the main purposes of this assignment is to get you familiar with github. You will walk through the following steps to complete the assignment.

1. The first step is making sure you have a github account. If you do not already have one, you may go to https://github.com/ to create one. Feel free to use your @email.arizona.edu email for the account, but if you already have an account that is not associated with your school email that is fine. **PER THE LECTURE SLIDES, MAKE SURE TO LET US KNOW YOUR GITHUB ID VIA THE GOOGLE FORM. IF NOT DONE, THIS MAY RESULT IN A REDUCTION OF POINTS. GOOGLE FORM**

2. Once you have your account, you must get your own private repository for this assignment. For each assignment in this class, you will get a private repository that you will submit your work through. That means you must create a repository for each assignment (don't worry, it's simple).
doing?
To get your own private repository for a given assignment, visit the github class room link for the associated assignment. For assignment 1, its the following link:
https://classroom.github.com/a/z7vS9OT-.

Make sure you are logged into github before clicking the link. When you click this link, you will be brought to a page asking you to accept the assignment. Click the “Accept this assignment” button and your repository will start to be created. You will be brought to a page saying the operation was successful and it will have a link to your repo. This link is located on the line that says “Your assignment has been created here:”. If you click that link, you will be taken to your repository.

3. At this point, you should have your own private repository. It should have one file inside of it called questions.txt. This file has questions you will answer, but we will get into that later. At this point, you need to clone your repository to your computer. Copy the link to your repository from the github page. Run a git clone command from your command line with the repository link to make a clone on your computer.

Note: The questions.txt file is worth 24 points, each question is worth 2 points (including filling out the info at the top of the file!).

4. At this point, you should have your repository cloned onto your computer. Go inside of your repo directory. There should already be one file inside of it called questions.txt. In this file there are questions that you need to answer. Open this file with your editor of choice, fill out the questions, save the file, and close your editor.

5. At this point you need to commit your changes and push the file to your repo. Use the “git commit -am” command to commit with a message. Then, use “git push origin master” to push your changes up to the github repository.

6. For the next step, you will add some files created by yourself to the repo. These files will be python files detailed below in the second section of this spec. Create these programs and move the files into your repositories directory.

7. Remember, just having the files within your repo’s directory doesn't mean they are being tracked by git, you must actually add them using the “git add” command. Once you have created the programs, use a “git add” command to add these files into the repository. At this point you can run “git status” and you should see your files have been added to the repo and are waiting to be committed. Now, as we did earlier, commit and push.

8. That's it! You can check your repository via the github web interface, but your filled out questions.txt and python programs should be held within it.

Specification Part II: Python Programs

For the second part of this assignment, you will be implementing several python programs that mimic command line tools. This will help you exercise your skills using stdin/stdout and command line arguments. Although it is not required, you are highly encouraged to use a text editor to create these programs.

Program 1 (6 points), echo.py

For this problem, you will write a python program that behaves like the echo command. It should take input as command line arguments and output whatever it is given there.

Example:

```
% python3 echo.py hello this is a test
hello this is a test
```

Program 2 (10 points), wc.py

This program will behave like the **wc** command, which counts the number of occurrences of newlines, bytes, and words within a file. For this program, you will read in the file name you are evaluating from the command line. You also may be given options as command line arguments. If given the **-w** option, you should only report the number of words. If given the **-b** option, you should only report the number of bytes. If given the **-n** option, you should only report the number of newlines. You will only ever be given one option if any are given. The option will only ever appear as the first argument.

Example:

```
% cat in
back then
a cow
came to me
% python3 wc.py  in
3 7 27
% python3 wc.py  -w in
7
% python3 wc.py  -n in
3
% python3 wc.py  -b in
27
```

Program 3 (10 points), cat.py

For this program you will be implementing the cat command. Cat works by taking any number of filenames on the command line as arguments and printing out their contents. If no filenames are given, take input from standard input. You should implement two command line options. There are two possible options, -n or -b. Look at the cat man page and determine what -n and -b do. The behavior in your program should be identical to the cat UNIX command when given those options. An option will only ever appear as the first argument and only one will ever be used at a time.

Example:

```
% cat in
this is the contents of in
% cat test
this is the contents of test
% python3 cat.py “in” “test”
this is the contents of in
this is the contents of test
```

Miscellaneous

The point breakdown for this program is as follows:

Running your submission using the following command**, python3 *yourprogram***, should run your program without error.

This assignment is submitted through github. This may be a new idea to some students, so make sure you understand how it works.

Remember, do not cheat! Refer to the syllabus and first lecture for more information.