COMP 206 - Winter 2018 - Assignment 0 (edited Jan 17 with the Github work-around)

Goal to provide an introduction to the programming environment we will use during the term, ensure you have the correct accounts and access needed to use the McGill computing resources we will use for the rest of the assignments.

Not for marks, but required as you need access to GitHub to get the course material and to know how to submit to My Courses to complete the assignments.

Due date, this is informal since it's not for marks, but we'll give the feedback on your submission on Jan 23rd, 2018.

Question 1 - Our Course GitHub

Create your account on McGill's Github. This site is only accessible from on campus, or through McGill's VPN. Instructions for VPN are here VPN Instructions. Now, open https://github.mcgill.ca in your browser. Log in with your McGill Campus Wide Login (the one used for Minerva and My Courses). As long as you're registered for the course or the waitlist, you should have the ability to create an account.

Find the repository for this assignment: https://github.mcgill.ca/COMP206Winter2018/Assignment0. Take a look at the files through the Github web interface. You will be adding your own file to this repository using the Linux shell to operate git from the command-line following the "clone, edit, add, commit, push" cycle described during Lecture 2. There are several steps, so follow closely as we walk you through:

Step 1: Add your ssh keys to GitHub, which is required so that the server can associate your Linux user account with your McGill credentials:

1. Run "ssh-keygen" in your Linux shell. You can leave your passphrase blank and just keep pressing enter until the process ends (or choose a passphrase if you wish, it is slightly more secure)
2. Select settings from the menu at the top-right of the Github webpage
3. Enter the SSH and GPG keys section by selecting that option from the left panel
4. Click "New SSH Key" on the top right of that page

5. Open the file `~/.ssh/id_rsa.pub` in Linux (with a text editor, or by using the cat command) and copy the contents
6. Paste the file contents into the box on the Github webpage
7. Give your key a name you can remember, such as "SOCS"
8. Click add SSH key

Step 2: Clone the repository, which gives you a local copy of all the files:
1. Type "git clone git@github.mcgill.ca:COMP206Winter2018/Assignment0.git" into the Linux terminal
2. You should now have an Assignment0 folder in that directory, enter it by typing "cd Assignment0" into the terminal
3. View the files in that folder with "ls", "cat", or by using your favorite text editor

Step 3: Add a new file to the directory, using a text editor.
1. The response file you add should follow the specification that you can find in my response, david_meger.txt. That specification defines that the file name is your first and last name separated by an underscore, with the ".txt" extension. Copy the first line of david_meger.txt into your file, so it's identical. Write you own content for the 2nd line, making sure to separate each answer using commas in between them. You should not use any commas within the answer for each question, since the commas are used to separate answers.
2. Note that you are not required to provide your real name, since this repository is public and anyone with access to this GitHub server will be able to see our responses. Later when we submit code for real assignments with grades attached, we will use private repositories that are secure. Also, the answers for each question are just for fun, so feel free to make something up, or leave a blank simply by having only spaces between two commas.

Revised Step 4: Submit to My Courses We have discovered a new limitation with McGill's GitHub that prevents us from easily giving you all the permissions to push to the Assignment 0 repository. For now, please hand in your <name>.txt file along with the answer from question 2 to My Courses. We will later upload the files to GitHub for you so the end result will be the same. For your information, the instructions below are how developers publish changes if they have push access (and we will do this later in the course).

The step we planned, but that does not work: Add, commit and push your new

file, sends the local data up to the cloud server:

1. "git add <your file name>"
2. "git commit -m "<your log message>"
3. "git push" (this will always give you a permission denied error now, which is the issue we discovered. So, no need to go further)
4. FYI, if you have correct permissions, there would be possible outcomes of push:
   1. The push completes without error, you are done and can now see your file through the Github page
   2. The push outputs text like "error: failed to push some refs... hint: Updates were rejected because the remote contains work that you do not have locally". This means another student pushed in between your clone and push. You need to get a fresh copy and push again, with these commands:
      1. "git pull"
      2. "git merge"
      3. "git add <your file name>"
      4. "git commit -m "<your message>"
      5. "git push"

Question 2: Hello, world

Ensure that you can create a file named "hello.c", compile it using the command "gcc hello_world.c" and run with the command "./a.out" to see "Hello, world." printed on the terminal as was shown during Lecture 2.

Once your code works by running manually, use our automated tester, which is found inside the same Assignment 0 repository you cloned in Question 1, within the "eval" folder. Run the tester by typing "python eval.py <path to folder that contains your code>". You will see output on the terminal indicating the outcome. We will often provide you testers because they help you ensure you match the **coding specification** perfectly. Once your code passes both of the 2 tests, it is quite likely close to a good solution. Do note though, we will run additional private tests during marking, so it is not sufficient for you to simply read the tester and write code that only passes those exact cases. You must always think about writing code that is robust and will satisfy the specification for all cases requested.

Submit your hello.c file, as well as your <name>.txt file from Q1 on My Courses, under

the Assignments tab and in the Assignment 0 Handin folder. This assignment is not marks, but we will run our automated testing on all code submitted, and in case we find any problems with the way you submitted, we'll let you know to prevent problems later.