Tutorial 06 - Collaborating with Github

why git and Github???

- version control (`git`) allows you to keep a record of development of code
- having multiple versions allows for testing new functionality or analyses, but provides an opportunity to return to a previous version if new functionality doesn't work as planned
- `Github` is the remote (or online) version of `git` and allows for sharing and collaboration with others when developing code
- we will use `Github` for the remainder of the semester to assign and collect exercises

git commands vs. shell commands

- git commands are different from shell commands
- shell commands are components of `bash`
- someone wrote code (C, shell, Perl, and Python) that created a collection of functions, including `init`, `add`, `commit`, `push`, and `pull`
- we can execute these functions inside `bash` using the program `git`
- next week you will see some other examples of programs written in other languages that we can execute in `bash` to do bioinformatic analyses

add vs. commit

- `add` includes the current version of files in the staging area
- `commit` creates a snapshot or time capsule of all files in the staging area
- this two step process allows us to only record information abou files we've made changes to and added to the staging area in a commit call, rather than always keeping track of all files
- it also allows us to be sure what changes to what files we want to commit before doing so

commit messages

- commit messages are like comments in your code
- commit messages are for you or future users of the repository to understand the changes that were made to the code over its development
- `git` forces you to "comment" each of your commits because it is a good practice

local vs. remote

- although these concepts apply to version control - `git` works locally and `Github` works remotely - this is a more general concept in computing
- local refers to work on your individual computer; all of our work previous to this week has been locall
- remote refers to work on a distant computer using an internet connection

ssh

- `ssh` is a secure shell that connects a client (you on your local machine) with a remote internet

-`ssh` is used to make a public, unsecured communication channel (the internet) secure

- `ssh` communication is secure because the information sent between client and server is encrypted (sent using a complicated secret code)
- key pairs between the client and server limit access to a specific client who has the correct password or a digital key that matches the other half of the pair contained on the server

ssh and Github

- recently `Github` has adopted more security for users
- `ssh` is one method being used to provide this elevated security
- to use `Github` securely you must generate a key pair; half of the key pair is stored on your local computer and the other half is placed on your `Github` profile

common sequences of commands with `git` and `Github`

starting a repo from scratch:

- on the `Github` website create a new repository and copy the address for the new repository (be sure to select ssh)
- `clone` the repository on your local machine using `git clone`
- develop code locally and generate versions as you work using `git add` and `git commit`
- send your local code to the remote `Github` repo using `git push`

common sequences of commands with `git` and `Github`

borrow someone else's code and improve yourself

- on the `Github` website create your own copy of someone else's repo by `forking` it
- `clone` the repository on your local machine using `git clone`
- develop code locally and generate versions as you work using `git add` and `git commit`
- send your local code to the remote `Github` repo using `git push`
- if you wanted to share your new code with the original owner you can submit a `pull request` on the `Github` website

How to do an assignment from now on

1) fork the TAs repository

2) clone the github repo to your local machine

3) do your work; during which time you can add and commit changes locally as you wish; you can also push changes to your repository as you wish

4) do one last add and commit to record all of your changes

5) push your local git repository to your github repository

6) submit a pull request on Github in a web browser by 10:30 on Fridays to "turn in" your answers

Exercise 6

- starts with a walkthrough exercise to replace SWC collaboration activity
- setting up Github `ssh` help for those that need it
- everyone must turn in their own via a pull request