Git Workflow:

* To create a new repository
  + navigate to github.com
  + click “New Repository”
  + fill out the name and readme
  + click create repository
* To create the local repository on your computer
  + Navigate to the main page of the repository
  + Click “clone or download” (a pull down menu will appear)
  + Copy the link in the box on the pull down menu
  + Switch to your command line program
  + Navigate to the location that you would like to clone the repository into using cd and ls
  + Type “git clone <the link that you pasted>”
  + A folder should appear
* To get someone else’s repository
  + Go to the main page of the repository
  + Press fork (top right)
  + You will be asked where to fork it to, select your profile
  + Then follow steps of creating local repository above from the fork that is now on your profile
* Updating your repository
  + Once you have a folder for your git repository, interact with it normally as you would any other folder on your computer
  + To check whether you have updated any files in your local repository, navigate to it in the command line using cd and ls
  + Type “git status”
  + Any file files that have been updated will be listed in red
  + For any files that you have changed and would like to update on github, type “git add filename” (use tab complete for this)
  + Type “git status” to check if git add worked; if it did, the file that was typed in should now be green
  + Type ‘git commit –m <some description about the update>’
  + Type “git branch –v” to see which branch you’re on. It should return something like this: \* master d146302 updates, where master is your branch
  + Type “git push origin master” (last word is your branch name) and your updates should be pushed to to the origin, which is the repository on github
* Branching
  + Create a new branch “git branch <branch name>”
  + Delete a branch once it is merged “git branch –d <branch name>”
  + Rename git branches “git branch –m <branch name>”
* Create a github repository from an existing folder
* Pull Requests
* Merging

Git ignore: to not upload all files on the repository to github

* Create an invisible folder in your git repository for your ignored files by navigating into your repository in command line and typing “touch .gitignore”
* Add in a file or folder to ignore by typing “echo ‘foldername/filename’ >> .gitignore”
* Or

Remove files from project directory, without actually deleting them (not tested yet)

* Remove file from project directory by typing “git rm --cached foldername/\*” (asterisk denotes all files in a folder
* Put the file into your .gitignore by following the steps in the preceding section
* Stage the file to commit by typing “git add .gitignore”
* Commit and push

Pull Requests

* **Step 1:** From your project repository, bring in the changes and test.
* git fetch origin
* git checkout -b Ben origin/Ben
* git merge master
* **Step 2:** Merge the changes and update on GitHub.
* git checkout master
* git merge --no-ff Ben
* git push origin master