2019-2-4

Lab 06 - Due at end of class

Sorting, Merging and testing jointly developed code.

Work with a partner. Each of you will implement a sort function of the form

def LastNameSort(alist):

...

...

return sortedList

You will work with a randomly assigned partner, and be given a new assignments repo to work from for just that group. Recall the steps to begin working with a new repo. You will need to clone that repo to your local machine, and do all your work there. You will not be working in your previous assignments subdir (ie. Assignments-github\_username), instead you will be working in a new subdir that is assigned the night of the lab. I would recommend naming it something meaningful, such as assignment-lab6.

To do this, when you clone it, you can provide an optional subdir name when cloning.

Ie. git clone <https://github.com/nwu-cs/assignments-group1.git> assignment-lab6

You can also just rename the subdir to whatever you like after you clone from github

1. Make a branch in your local repo called dev\_YourLastname

**Git checkout -b dev\_YourLastname**

1. As an individual, you will be assigned a random sort routine from the page

<http://interactivepython.org/runestone/static/pythonds/SortSearch/toctree.html>

Write your randomly assigned function in a file named “**Lastname**Sort.py”, replacing your lastname in the filename. Start with the unittest template I provided, unittest.py and modify it to import and call your new sort algorithm. Once you are satisfied it works, commit your changes and push to github

1. After both you and your partner both finish, push your code to github, and once your partner has also pushed, pull/update your local repo from github.
2. You should now see in your git client your partner’s branch. You want to merge in their code into yours. Do this with “git merge origin/dev\_partnersName” or use the merge command from your git client. There will likely be some conflicts. Git status will say which file has the conflict. Open that file.

The file will be modifed with both sets of changes in it, and several markers as “>>>”, “===”, and “<<<” . The markers show where the conflicting changes came from. Everything between the <<<<< HEAD and ===== are YOUR local changes. Everything between the ==== and >>>> origin/merge-exercise are the REMOTE's changes (e.g. your partner's). Just modify the file to look the way you want, and save. Then stage/add and commit your changes.

You want to make changes so that the test\_sorts.py file will call both your sort function and your partners, and tests them using the assertEqual function.

1. Push your changes to github and create a pull request, assigning it to me.