30538 Problem Set 3: git

Peter Ganong, Maggie Shi, and Dema Therese Maria Palathingal

10/21/24

Due Sat Oct 26 at 5:00PM Central. Worth 50 points.

Before you begin this problem set, you will need to install command-line git

Your submission does not need runnable code. Instead, you will tell us either what code you ran or what output you got.

2. Late coins used this pset: **___** Late coins left after submission: **___**

1 Solo

(applies only to the solo part) "This submission is my work alone and complies with the 30538 integrity policy." Add your initials to indicate your agreement: **___**

problem set setup (5 points)

1. "I have uploaded the names of anyone I worked with on the problem set **here**" **___** (1 point)
2. Knit your `ps3_solo.qmd` as a pdf.
3. Push `ps3_solo.qmd` and `ps3_solo.pdf` to your github repo. Use command line git (not github desktop)
4. Submit `ps3_solo.pdf` via Gradescope (4 points)
5. Tag your submission in Gradescope

1.1 Learn git branching (15 points)

Go to https://learngitbranching.js.org. This is the best visual git explainer we know of.

1. Complete all the levels of main "Introduction Sequence". Report the commands needed to complete "Git rebase" with one line per command.

2. Complete all the levels of main "Ramping up". Report the commands needed to complete "Reversing changes in git" with one line per command.
3. Complete all the levels of remote "Push & Pull – Git Remotes!". Report the commands needed to complete "Locked Main" with one line per command.

1.2 Exercises (25 points)

Now it's time to get your hands dirty! Clone https://github.com/eficode-academy/git-katas.git

Tips:

- These exercises have many steps. Keep a notebook (e.g. `.txt` or note-taking software) with what happens at every step.
- To find out what directory you are in, run `cd` on a PC or `pwd` on a Mac.

1.2.1 Basic Staging and Branching (5 points)

1. Exercise. For your pset submission, tell us only the answer to the last question (22).
2. Exercise. For your pset submission, tell us only the answer to the last question (18).

1.2.2 Merging (10 points)

1. Exercise. Report the answer to step 12.
2. Exercise. Report the answer to step 11.
3. Include a picture (on the computer or take a photo of a pencil and paper picture) of the commit history in exercise 1 and exercise 2. In words, explain what is the difference between a fast-forward merge and a 3-way merge.

1.2.3 Undo, Clean, and Ignore (10 points)

1. Exercise. Report the answer to step 13.
2. Exercise. Look up `git clean` since we haven't seen this before. For context, this example is about cleaning up compiled C code, but the same set of issues apply to random files generated by knitting a document or by compiling in Python. Report the terminal output from step 7.
3. Exercise. Report the answer to 15 ("What does git status say?")

2 Partnered

2.1 Git merge conflicts (10 pts)

For this problem set, you and your partner will accept the same partnered assignment using this link XXXX

You do not need to turn in any `qmd` or `pdf` as part of this section. You will be graded based on the commit history from doing the steps described below.

Name of your partner:

Prelude

i. Play paper, scissors, rock to determine who goes first. Call that person *Partner 1*.
ii. Both partners clone `ps3_pair`.

First round of practice

i. a. *Partner 1*, Start a branch called `merge_conflict_name_1`. In `ps3_pair.qmd` replace "Dema Therese" with your name. Push your branch to github remote
   b. *Partner 2*, Start a branch called `merge_conflict_name_2`. In `ps3_pair.qmd` replace "Dema Therese" with your name.
ii. *Partner 1* screen share and make a pull request on github.com.
iii. *Partner 2* screen share on github review the pull request. Accept your partners changes and merge the branch into `main`. Hooray! This is your first successful pull request!
iv. *Partner 2* make a pull request.
v. *Partner 1* screen share. On github.com review the pull request. There should be a merge conflict because you both changed the same line of the file. Adjust the file and then complete the merge.

Second round of practice

i. a. *Partner 2*, Start a branch called `describe`. In `ps3_pair.qmd`, modify the function so that it returns a list where the first object is the material printed as the head and the second object is the results from running `describe` on the data frame
   b. *Partner 1*, Start a branch called `histogram`. In `ps3_pair.qmd`, modify the function so that it returns a list where the first object is the material printed as the head and the second object is an altair plot with a histogram of the values
ii. *Partner 2* screen share and make a pull request on github.com.
iii. *Partner 1* screen share on github review the pull request. Accept your partners changes and merge the branch into `main`. Hooray! This is your first successful pull request!

iv. *Partner 1* make a pull request.

v. *Partner 2* screen share. On github.com review the pull request. There should be a merge conflict because you both changed the same line of the file. Rewrite the function so that it returns a list with three objects (`head`, `describe`, and histogram) and complete the merge