Git Challenge Step 4

1 Background

Remotes are repositories that are associated in some way with a given repository. Each remote has an (essentially) arbitrary name and has associated URLs telling git where to fetch and push data. The `git remote` command has various options and subcommands that let you examine and change remotes.

The most common case is when our repository on our local machine is tracking a version of the repository on some public- or group-facing server (like github), which acts as an "upstream" repository. Changes in the remote repository are fetched/pulled down to the local machine to keep the local repository in sync, and changes in the local repository are pushed up to the server. This allows many people to collaborate by working independently on the code and merging in changes that others make. The two repositories differ but are kept roughly in sync over time as changes are pushed and pulled in both directions. This is what happens when you clone a repository on github. Your local clone has the github repository as an associated remote (called `origin` by default).

But in general, one may have many remotes associated with a repository, offering the possibility of pulling code (or pushing out code) from or to various collaborators. This is part of what makes Git an effective *distributed* system for version control. There need be no central repository; any group of repositories can be kept in sync.

2 Tasks

You should still be on the `master` branch after the last step.

- Look the remotes for this repository using the `git remote -v` command.

- Having cloned this repository, you should see the `origin` remote listed, with a URL for both fetch and push. You do not have push permission on the `git-challenge` repository on GitHub, so GitHub would not allow you to push to it.
- Now, use the `git remote` command to associate a new remote repository `https://github.com/36-750/git-challenge-supplement` from which you can obtain code. (Hint: use the `git remote add` subcommand.) Call the new remote `supplement`.
- Use the `git remote show` command to see details about the `supplement` remote.
- Fetch this remote into your repository with the `git fetch` command. You will need to specify the remote at least. Be sure to fetch *all* of the data from the remote, not just the `supplement/master` branch.
- Use the `git branch` command to list all remote branches in the repository. (Look at the available command line options for `git branch`.)
- Find a branch in `supplement` named after one of the planets in the solar system.
- Use the `git branch` or `git checkout` command to create a local branch that "tracks" this remote branch.
- Move to this local branch and use `git pull` to populate your new branch with the remote data.
- The file `NEXTSTEP.pdf` in this branch has the instructions for the next step; the branch's name is the password for the file.

3 Resources

- Basics of Remotes `https://git-scm.com/book/en/v2/Git-Basics-Working-with-Remotes`
- Documentation (type `git help COMMAND` or follow the links below) for
  - `git remote` to move between branches: `https://git-scm.com/docs/git-remote`
  - `git fetch` to move between branches: `https://git-scm.com/docs/git-fetch`

- `git pull` to move between branches: `https://git-scm.com/docs/git-pull`
- `git branch` to move between branches: `https://git-scm.com/docs/git-branch`
- `git checkout` to move between branches: `https://git-scm.com/docs/git-checkout`