Git practice

1. Clone the class repository
    a. Create a directory called class_repo.
    b. cd class_repo
    c. Initialize the directory as a git repository.
    d. git init
    e. Clone the class repository.
    f. git clone https://github.com/kmaloneVillanova/2053Spring23.git

2. Pull changes from class repository (GitHub)
    a. Make sure you're in the correct local directory – git/class_repo/2053Spring23
    b. git pull origin main
    c. Note on the last command. Some GitHub branches are called main by default, others are called master. You need to issue main or master based on what your default branch is called. To check, look at the command prompt. On GitBash it will have the branch name at the end of the command line in parenthesis. If you don't' see it, type git branch and it will return the branch name. Use that name to push to GitHub.

3. Create your own repository
    a. Go to github.com
    b. In top left where it says, "Top Repositories", click New.
    c. Give your repository a descriptive name.
    d. Keep the default Public
    e. Check "Add a README file"
    f. Back on your laptop, create a directory called my_repo.
    g. cd my_repo
    h. Initialize the directory as a git repository.
    i. git init
    j. Clone your repository. Get the repository URL by going to the repository on Github and clicking on the Green code button. The drop down menu will show you the repository URL.
    k. git clone <enter repository URL here> (do not include the <>)

4. Add files to your repository and push to GitHub.
    a. When you're ready to push to GitHub issue the following three commands from the repository root directory. If you're working with the repository you made in class, go to git/my_repo/CSC2053
        i. git add .
        ii. git commit -m "comments about what you added or updated"
        iii. git push origin main

Note on the last command. Some GitHub branches are called main by default, others are called master. You need to issue main or master based on what your default branch is called. To check, look at the command prompt. On GitBash it will have the branch name at the end of the command line in parenthesis. If you don't' see it, type git branch and it will return the branch name. Use that name to push to GitHub.

5. Add collaborators to your repository
    a. Go to your GitHub repository and click Settings in the top right hand corner.
    b. On the left hand menu under Access click collaborators.
    c. Click Add People
    d. Add the collaborator's github username.
    e. An invitation will be emailed to the collaborator. They must accept the invitation to be added to the repository.

6. Resolving merge conflicts
    a. A merge conflict arises when two collaborators in a repository both update the same file. Before changes can be pushed to GitHub the conflict in changes must be resolved.
    b. First let's create a conflict so we can resolve it.
    c. Have collaborator 1 create a file called contact.txt with their contact information like name and email address. They should push the file to GitHub (see number 4 above for commands to do this).
    d. Have collaborator 2, pull the changes from GitHub (See number 2 above for instructions on how to do this).
    e. Collaborator 1 and 2 should both update this file with changes. Collaborator 2 can add their contact information and collaborator 1 can add their phone number.
    f. Collaborator 2 should push their changes to GitHub.
    g. After collaborator 2 pushes their changes, collaborator 1 should try to push their changes. Collaborator 1 will receive an error that there is a merge conflict.
    h. To resolve the merge conflict, collaborator 1 should pull changes first.
    i. After they do a pull, the changes from collaborator 2 will be added to the contacts.txt file. Collaborator 1 should open the file and remove the headings added by GitHub. They should review the file and make sure it is in the condition they want.
    j. Now push the changes to GitHub.