Homework 1

Before you begin!

⟹Note that if you try to do this homework in any way other than what I have described it could be counted as CHEATING! This is an individual assignment!

⟹I have added some helpful links a the bottom of this assignment to help guide you with your assignment and to act as supplemental material.

Purpose

Create and get familiar with GitHub's online website and the Git version control system on the command line.

1 GitHub Account Creation

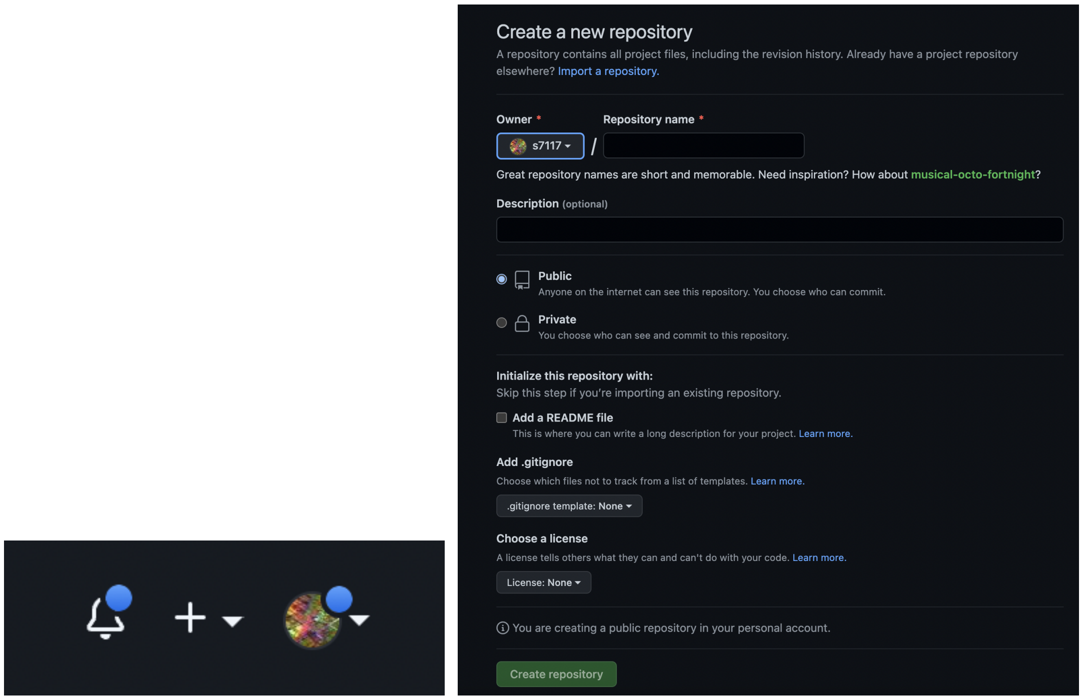
Create a GitHub account by going to `https://github.com` and clicking on "Sign up for GitHub" after entering your personal or university email (I suggest your personal email because you may want to use this outside of college one day!).

Go through the setup process and then you can continue the assignment once you have an account.

2 Your First Repository!

To create your first repository click the + symbol in the top right and choose "New repository" (see Figure 1).

Figure 1: Creating a new Git repository.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner * / Repository name *

s7117 /

Great repository names are short and memorable. Need inspiration? How about musical-octo-fortnight?

Description (optional)

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☐ Add a README file
This is where you can write a long description for your project. Learn more.

Add .gitignore
Choose which files not to track from a list of templates. Learn more.
.gitignore template: None

Choose a license
A license tells others what they can and can't do with your code. Learn more.
License: None

You are creating a public repository in your personal account.

Create repository

You will first need to create a GitHub account at https://github.com if you do not have one already. Once you have an account you should be able to navigate to *https://github.com/<your_user_name>* to see your profile.

After clicking "New repository" you will be brought to a repository setup screen seen in Figure 1. Follow these steps:

1. Enter '.dotfiles' as the "Repository name". Be sure to include the '.' at the beginning!
2. Enter 'My shell dotfiles.' as the "Description".
3. Ensure that "Public" is selected.
4. Do **NOT** check the "Add a README" box!
5. Leave the "Add .gitignore" section default to 'None'.
6. Leave the "Choose a license" to its default value of 'None'.

Now you should have an empty repository that we can add in the next steps.

3 Setup Git on the Command Line

To setup Git on the command line you will need to run two config commands:

```
git config --global user.name "Firstname Lastname"
git config --global user.email "yoursignupgithubemail@domain.com"
```

Now comes the more difficult part: Go to the ".ssh" directory in your home directory '~'. If the ".ssh" directory does not exist in your home directory then create it and then change to it. Now that you are in the ".ssh" directory do the following:

1. Create a file named "config".
2. Edit the file with the exact text below:

   ```
   # GitHub.com
   Host github.com
       Preferredauthentications publickey
       IdentityFile ~/.ssh/github/id_ed25519
   ```

3. Now save and quit the file.
4. Now create a new directory called "github".
5. Go into the directory "github".
6. Run the following command:

   ```
   ssh-keygen -t ed25519 -a 203 -f ./id_ed25519
   ```

7. When prompted for a passphrase you can (optionally) enter one. Just keep in mind that you will need to remember the passphrase for every time you wish to upload to GitHub.
8. Press enter multiple times until your prompt reappears.
9. Now list the contents of your current directory. You should see two files: "id_ed25519" and "id_ed25519.pub".

10. *****WARNING***** Do **NOT** share the "id_ed25519" file with anyone! It is like a password that you should never give out and will give someone **DIRECT ACCESS** to do anything they wish with your GitHub repos!

11. The ".pub" file is safe to share! Now, cat the "id_ed25519.pub" file. It should look like this with a bunch of characters where the ellipsis is:

```
ssh-ed25519 ... <uofsc_username>@cocsce-l1d43-##
```

12. Copy the output of the file to your clipboard i.e. select the text and copy it using right click and then copy.

13. Now go to github.com and navigate to your profile "Settings" using the profile icon in the top right corner.

14. You should see a list of settings on the left side of your screen. Choose "SSH and GPG keys".

15. Create a new entry by clicking "New SSH key".

16. Set the "Title" to "Linux Labs".

17. Now right click and paste the cat output of the id_ed25519.pub file.

18. Finally, click "Add SSH key".

In this section we setup Git's configuration and we created an SSH key which can securely upload your code and stuff to GitHub.

4 Using Git!

You MUST record all of the CORRECT commands you use for this part.

Add the commands to a file named username_hw01.txt using VIM.

So now you should have a GitHub account and should have Git setup on the command line in Bash! Now we need to actually use it and see its POWER! We will first create a directory to host our Git repository. Then do several things in that directory. Finally we will upload those things to our Git repository on GitHub so they can take advantage of version control!

4.1 Create stuff locally

Be sure you are in your HOME DIRECTORY!

1. Create a directory named ".dotfiles". Don't forget the '.' in the front! This makes the directory a hidden directory!
2. Confirm you created the directory by listing **all** the contents and making sure it is listed.
3. Now change to the ".dotfiles" directory.
4. Confirm you are in the ".dotfiles" directory by printing the working directory.
5. Now create a file named "README.md". The 'md' file extension stands for markdown which allows you to somewhat code the format of your document via simple symbols.
6. Open the file using the text editor that shall not be named! :) ←Smile
7. Now you should have the "README.md" file open and be ready to add stuff to it.
8. At the top of the document, insert the following (you can copy and paste it in):

```
# My Dotfiles
These are my dotfile configuration files for different software in Bash.
## .nanorc
This is my custom .nanorc configuration for Nano.
## .bashrc
This is my custom .bashrc configuration for Bash.
```

9. Once you have finished adding the above lines save or write the "README.md" file.
10. Now you can quit the text editor.
11. Print the file out to the terminal to confirm the file was written to. You should see the six lines from above.
12. Now we are going to create a .nanorc file
13. Run the following command:

```
nano .nanorc
```

14. Inside of nano, put the following text

```
set autoindent
set boldtext
set mouse
set tabsize 4
set keycolor blue,red
```

15. Finally, copy your ".nanorc" configuration file we just made to your home directory. If you don't move this you won't see any differences in your nano configuration

4.2 Init the repo!

Now we need to initialize the repository and link it to GitHub. Use the following commands inside of the ".dotfiles" directory:

```
git init
git add --all
git commit -m "INIT commit"
git branch -M main
git remote add origin git@github.com:<your_git_username>/.dotfiles
git push -u origin main
```

After entering the last command you may be asked for the passphrase you set in Section 3. If not, then your first push to the repository should complete and the prompt will reappear!

4.3 See what happened!

Now all the stuff you did locally on the computer has been pushed to GitHub! That means its online and even backed up! Along with it went some meta-data about all the changes you made where and when! Lets take a look!

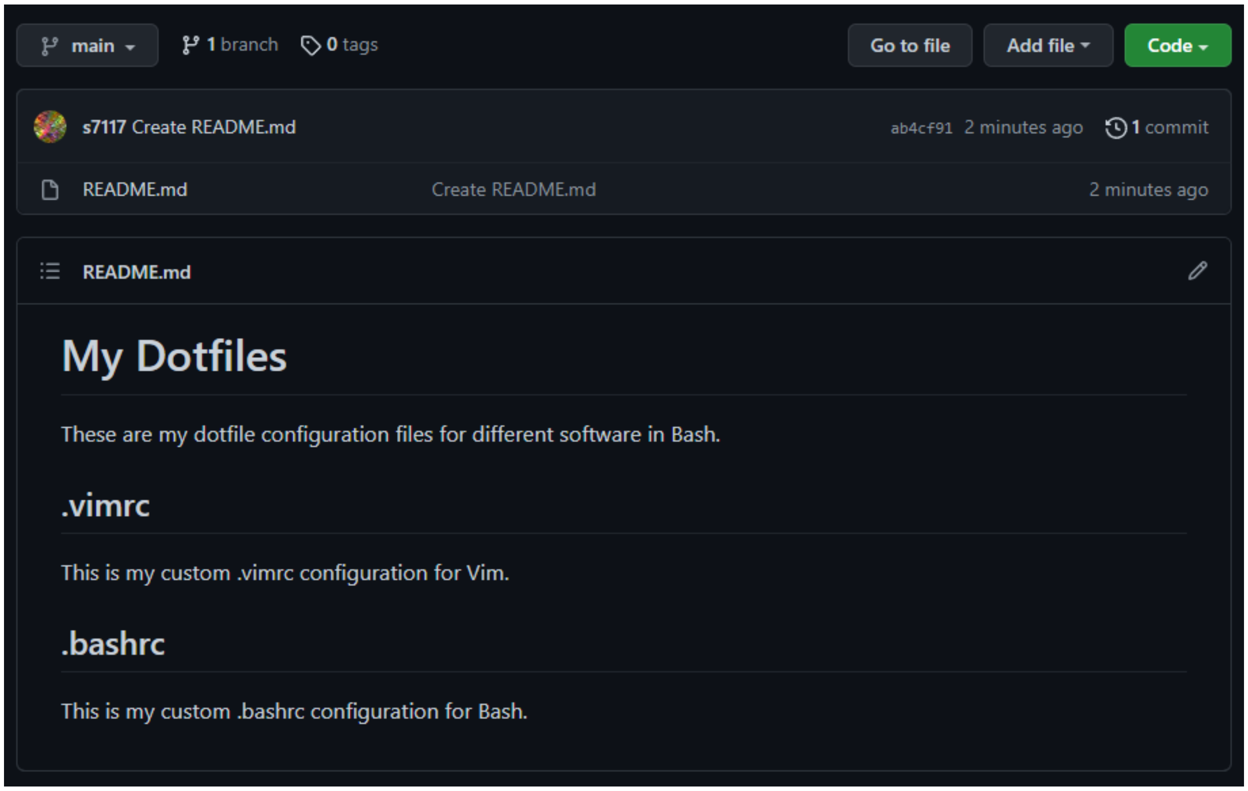
1. Go to your GitHub repository that you previously created. If you forgot where it is, it should be located at:

```
https://github.com/<your_user_name>/.dotfiles
```

2. Unlike before you should now see that "README.md" file and it should look really cool with headings and subheadings!

3. You should see something similar to the following picture in Figure 2 i.e. the README.md and the ".vimrc" file:

Figure 2: Repository root directory.



My Dotfiles

These are my dotfile configuration files for different software in Bash.

.vimrc

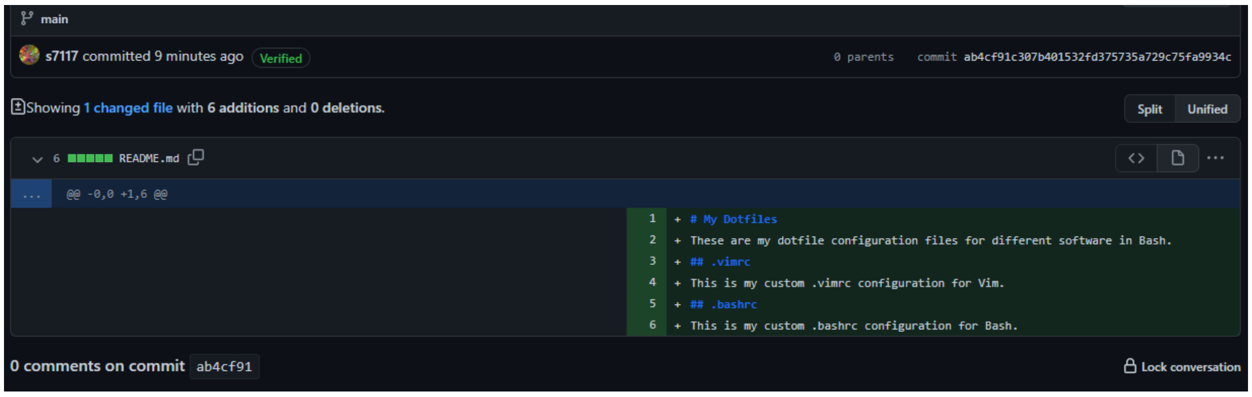
This is my custom .vimrc configuration for Vim.

.bashrc

This is my custom .bashrc configuration for Bash.

4. You should see a place that says "1 commit" below the green "Code" button. Click it!
5. This brings up a list of all the commits you have performed. Right now it is just one! Click on the name of the commit which should be "INIT commit".
6. Now you see all the changes! You can see that we created "README.md" and added six lines to it like in Figure 3.

Figure 3: Git diff of a commit.



main

s7117 committed 9 minutes ago Verified

0 parents commit ab4cf91c307b401532fd375735a729c75fa9934c

Showing 1 changed file with 6 additions and 0 deletions.

6 README.md

```
@@ -0,0 +1,6 @@
+ # My Dotfiles
+ These are my dotfile configuration files for different software in Bash.
+ ## .vimrc
+ This is my custom .vimrc configuration for Vim.
+ ## .bashrc
+ This is my custom .bashrc configuration for Bash.
```

0 comments on commit ab4cf91

5 Submission

1. To submit your work please create a **text** file named "hw1.txt" with the URL of your GitHub page in it.
2. After pasting the URL of your public GitHub repo at the top of this file you should write a small 3 to 4-paragraph summary of how you performed in the lab including any challenges you faced.
3. In this text file, you should also include all the commands you used to complete Section 4. If you forgot, refer to Section 6.
4. Finally you **MUST** cite all resources you used for your homework assignment.

6 Getting your history of commands:

If you didn't read the part about saving all the commands you use in Section 4, please do the following. Please be sure to carefully read the **ENTIRE** document before starting next time!

1. View the *.bash_history* file in your home directory.
2. Find where the commands where you started the assignment. Get the line number of the first command for the assignment.
3. Subtract the total number of lines in the *.bash_history* file from the line number for the first command of the assignment.

4. Use the ‘tail’ command to get all of the commands that were used in the assignment.
5. Redirect these to a file.

7 Helpful Resources

- GitHub’s guide to adding SSH keys:
  https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account
- Creating SSH Keys:
  https://www.unixtutorial.org/how-to-generate-ed25519-ssh-key/
- GitHub Markdown Cheat Sheets:
  https://www.google.com/search?q=markdown+github+cheat+sheet&rlz=1C1CHBF_enUS994US994&oq=markdown+github+&aqs=chrome.1.69i57j0i512l9.5975j0j7&sourceid=chrome&ie=UTF-8
- Git Cheat Sheet:
  https://education.github.com/git-cheat-sheet-education.pdf
- This is my .dotfiles on GitHub:
  https://github.com/s7117/.dotfiles