Introduction to Computing: Task 5
Fall 2018
Dr. Ramesh Yerraballi
Designed by Matthew Barondeau
The University of Texas at Austin
Due: Friday 11/30, 11:59pm

Purpose

Throughout your programming career, you will make many changes to your code and may wish to save key milestones in your projects. In addition, you may have to work in a collaborative environment where you are contributing code to a larger project. Version control systems allow you to track changes efficiently and manage your code in a larger project. One of the most commonly used version control systems is called Git. In this assignment, you will explore Git's many useful features so you can utilize them in your future coding classes.

Task Requirements

1. Create a pdf document titled README.pdf where you and your partner answer the following questions:
    a. Why is using a version control language useful in coding?
    b. Who developed Git?
    c. Define the terms: commit, push, pull, branch, merge conflict.
    d. What is the difference between git pull and git fetch?
    e. What does the alphanumeric string in a git commit refer to?
    g. Why are readme files important?
2. Individually create a public repository on GitHub
    a. Create a public repository on GitHub named EE306F18-GitTask (see below)
    b. Create a LC3 project that will compute the sum of two numbers and store the result at x4000. The inputs will be in x4001 and x4002. Start your program at x3000.
    c. Commit and push your project to GitHub.
    d. Restrict the program so that if either number is negative or zero, the program throws an error (sets result to -1)
    e. Commit and push your project to GitHub.
3. Work on Program 5 with your partner and push to GitHub
    a. Create a public repository and add your partner as a collaborator
    b. As you work on Program 5, push commits up to GitHub.
    c. Partners must alternate at least twice on commits. One partner cannot do all the commits, then the other does the last one. You must demonstrate pushing and pulling of code.
4. Git Report
    a. Prepare a pdf file documenting your development process for the two projects you created using Git. Describe the instances when you chose to commit, the commit/push/pull process (i.e. how you committed/pushed/pulled), and where you might use branches in your development process. Also include a description of how git blame works, and how it might be useful. Use screenshots of your code, the command line, and the repository where appropriate.

Grading

Individual Git Assignment	20
Partner Git Assignment	40
Question Section	20
Git Report	20

The Sections on the following pages are an introduction to Git Bash and GitHub. If you are already familiar with them, you are free to skip them. A more in depth guide can be found here: https://github.com/matthewbarondeau/GitGuide

GitHub Setup

To begin setting up GitHub, you will need to go to the following webpage: https://github.com/ and sign up for free. I recommend you use your utexas.edu email as it may give you some additional benefits. Once you have created an account, you can create a repository by clicking the green "New Repository" button as shown in Figure 1. You will need to make this repository public for this project as we want to be able to see it.

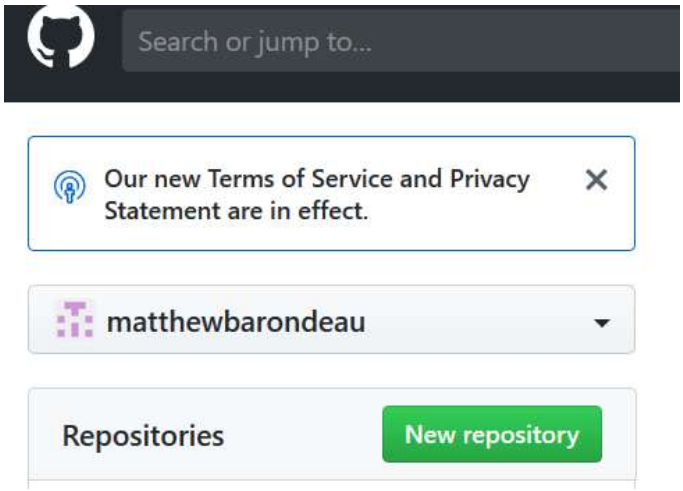


Figure 1. New Repository Button on GitHub

Git Bash Setup

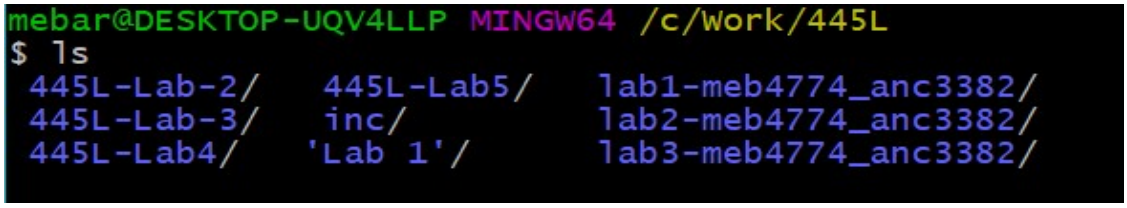
You now need a tool to interface with GitHub and use its functionality. GitHub provides a desktop application that you can use, but I suggest trying to use a command line interface. There is a useful command line terminal called git bash that is free to download. You can find git bash downloads located here: https://git-scm.com/downloads

Navigating Bash Terminal

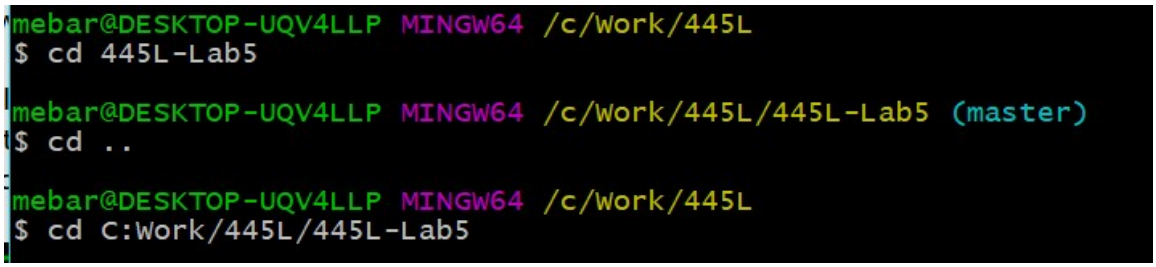
If you plan on using the GitHub desktop utility, you can skip this section. If you want to learn the bash utility or how to navigate a UNIX environment, I encourage you to read this section.

There are several commands that you will need to learn:

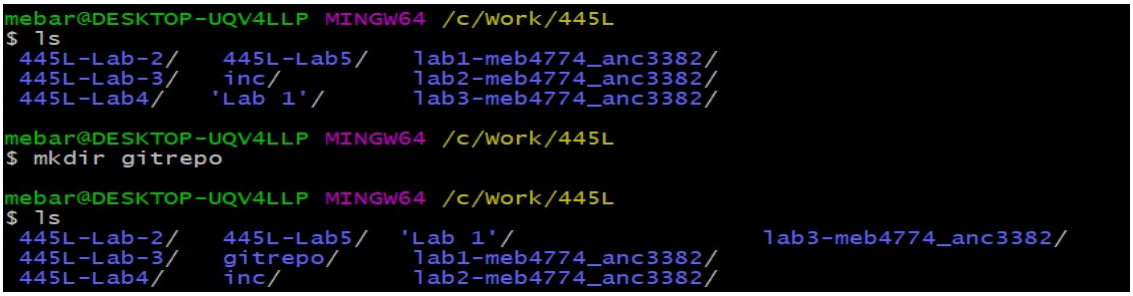
1. ls – Lists all the files in the current folder



2. cd – Changes folder (aka directory)
   a. cd .. – Moves "up" one folder
   b. cd [folder] – Moves to that folder ([folder] can be a path to a folder or it can be a folder in the current folder)



3. mkdir [foldername] – Makes a folder (directory) in the current folder



Try the following commands in order:

1. mkdir helloworld
2. ls
3. cd ..
4. cd helloworld
5. mkdir hello
6. ls
7. cd ..
8. cd ./helloworld

Git Concepts:

1. Cloning a repository
   Cloning is the process where you can link a GitHub repository (repo) to your host computer. You will do all your work in the repo on your host computer, then commit and push your work into the repo on GitHub.
   a. Create a new repo on GitHub.
   b. Get the address of the repo (where your repo is) by clicking the "Clone or download" button shown in Figure 5 and copying the link.
   c. Open Git Bash and type in "git clone [link]", replacing [link] with the link to the repo (See Figure 6). This will clone your repo onto your computer.
   d. Observe the contents of your directory with "ls" and make sure your repo is there as a folder. You can "cd" into the folder and then "ls" to see the contents of your repo.

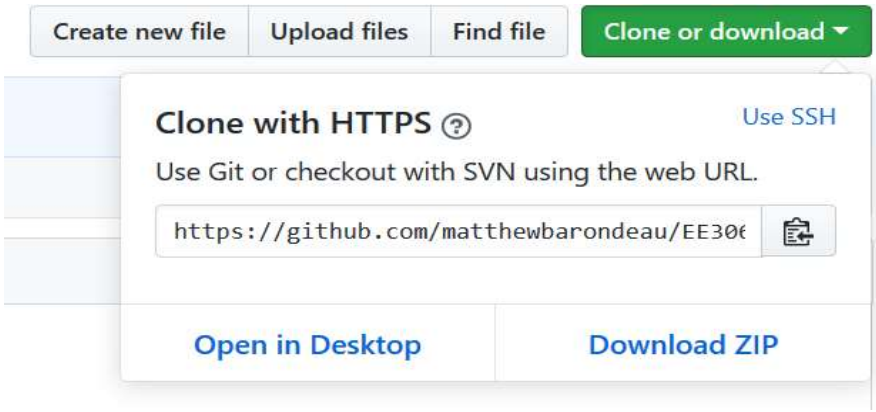


Figure 5. Clone or Download option on GitHub

```
mebar@DESKTOP-UQV4LLP MINGW64 /c
$ cd Work

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work
$ ls
 445L/   460M/   foldername/  'Learning Record MEB4744.docx'

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work
$ git clone https://github.com/matthewbarondeau/EE306-Task3.
Cloning into 'EE306-Task3'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work
$ ls
 445L/   EE306-Task3/  'Learning Record MEB4744.docx'
 460M/   foldername/    SnakeGame/

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work
$ cd EE306-Task3

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/EE306-Task3 (master)
$ ls
README.md
```

Figure 6. Cloning process using Git Bash

2. Making a commit to a repo
   Committing is the process by which you save important milestones. It consists of adding changes to a staging area, naming, and then saving those changes. I will then show you how to see what you have saved through git log.
   a. Make a non-empty file. In this instance it is an LC3 file that contains .ORIG and .END.
   b. Type “git status” to see changes since the last commit. Files that have changed since the last commit appear in red. Figure 7 shows an example of git status.
   c. In git, there are three main areas: The workspace, staging area, and the remote repository. When I change any file on my local machine, it is changing the workspace copy. When I decide that I want to save a file, I need to add it to the staging area. When I have everything in the staging area that I want to commit, I can make the commit and push it up to the repository. The saved copy will then stay in the remote repository.
   d. Type in “git add [filename]” to add the designated file to the staging area.
   e. Figure 8 shows that the status of git has changed. Green text indicates files in the staging area
   f. During a commit, all files in the staging area are added to the commit. You are required to state what each commit does. Use the following resource: https://chris.beams.io/posts/git-commit/
   g. Type “git commit -m [Message]” to perform the commit. See Figure 9.
   h. To see a list of previous commits, type “git log”. To exit the log type “q”. An example of git log is shown in Figure 10.

```
mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/EE306-Task3 (master)
$ ls
README.md  Task3.asm

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/EE306-Task3 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        Task3.asm

nothing added to commit but untracked files present (use "git add" to track)
```

Figure 7. Using Git Status

```
mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/EE306-Task3 (master)
$ git add Task3.asm

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/EE306-Task3 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git reset HEAD <file>..." to unstage)

        new file:   Task3.asm
```

Figure 8. Adding to staging area

```
mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/EE306-Task3 (master)
$ git commit -m "Add Task3.asm file"
[master 164e503] Add Task3.asm file
 1 file changed, 8 insertions(+)
 create mode 100644 Task3.asm

mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/EE306-Task3 (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Figure 9. Creating Git Commit

```
mebar@DESKTOP-UQV4LLP MINGW64 /c/Work/445L/445L-Lab9 (master)
$ git log
commit 55b2eca61cfa2b6a691115e5bd569659b78392f8 (HEAD -> master,
Author: matthewbarondeau <mebarondeau@utexas.edu>
Date:   Mon Nov 12 19:42:55 2018 -0600

    Temperature capture complete

commit 182d803cdaee2f1831578f91000d3bfbeae0c54a
Author: matthewbarondeau <mebarondeau@utexas.edu>
Date:   Sun Nov 11 14:54:14 2018 -0600

    Add interp function

commit 41239b8be9736a9d0bfa3528e5bee0ca74b39206
Author: matthewbarondeau <mebarondeau@utexas.edu>
Date:   Sun Nov 11 14:43:58 2018 -0600

    Add Graphics prototypes and edit main loop for temp readings

commit 6b9cc6c0390c4e594ec2ff56526e276d4d4e0488
Author: matthewbarondeau <mebarondeau@utexas.edu>
Date:   Sun Nov 11 14:03:16 2018 -0600

    Clean up main and add lookup tables
```

Figure 10. Git Log Example

3. Pushing to/pulling from a repo
   You have a local copy of the commits on your laptop at this point. This section covers synching up the repository on GitHub with your local changes. Note that you have to commit for a change to be pushed to GitHub.
   a. To push your commit, type “git push [location]”. In the example shown in Figure 11, I am pushing to the origin. This will normally be the location to push to.
   b. Sometimes you want to pull code from the repository to your local copy. This is done by typing “git pull [location]”. An example of a git pull is shown in Figure 12.
   c. If there are changes in the repository that you have overriden, you will end up with a merge conflict. You will need to manually select which code you want to keep if this occurs.
   d.

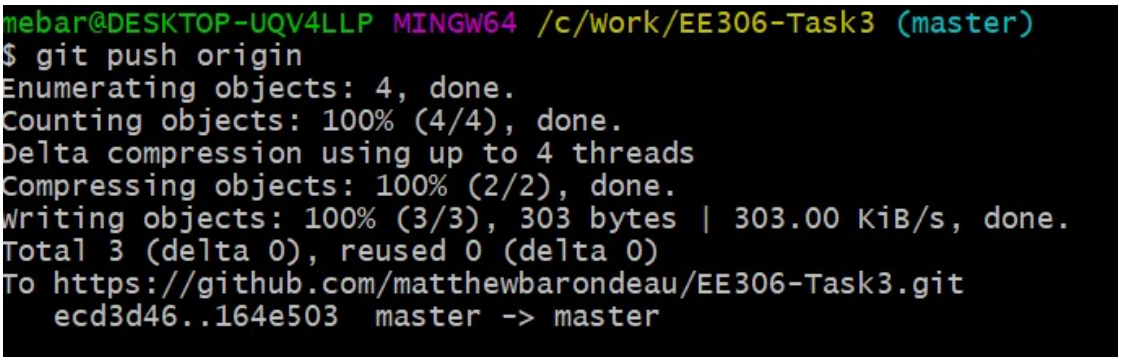


Figure 11. Git Push Example

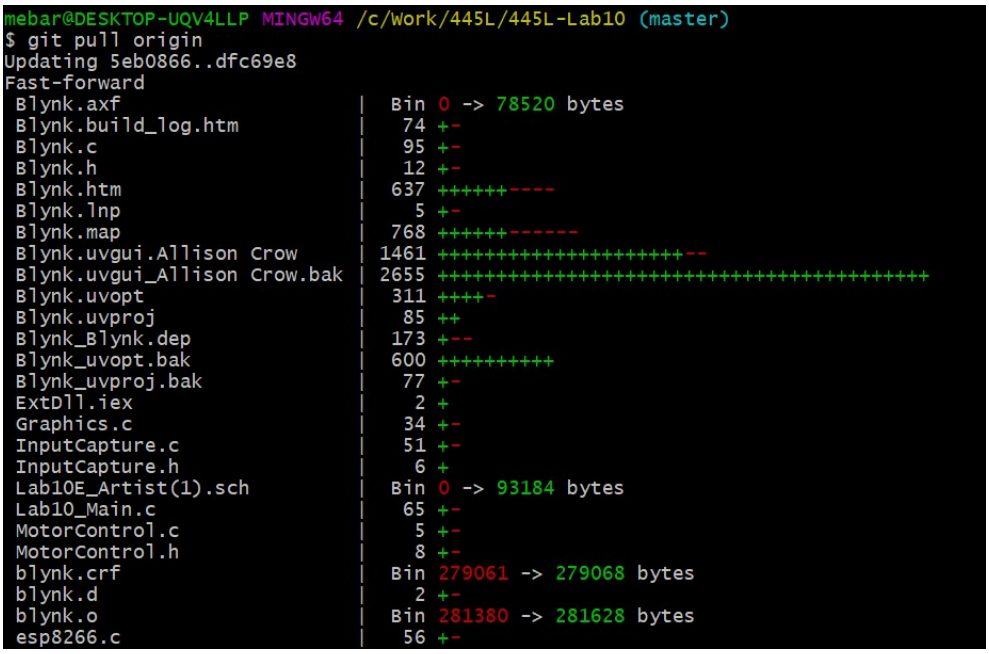


Figure 12. Git Pull Example