Lab Week 1

This semester you'll be using git to submit your assignments. Git is a source code control system for managing changes to files. We'll be using github classroom. For each lab assignment this semester your instructor will post a link that has the assignment. Employers want you to have a good understanding of Source code control and tools. This is your first step in that direction.

You may want to bookmark the git cheatsheet. https://education.github.com/git-cheat-sheet-education.pdf We will only be using a few commands this semester

Command	Action
git clone [url]	Clones the git repository to your location machine
git add [file]	Adds the file into to the commit stage. Anytime a file is changed you want to add it for the next commit
git commit -m "message"	Commits all the changes with the message given
git push	Pushes the current branch back to the repository.
git status	Returns the status of all files, controlled and uncontrolled

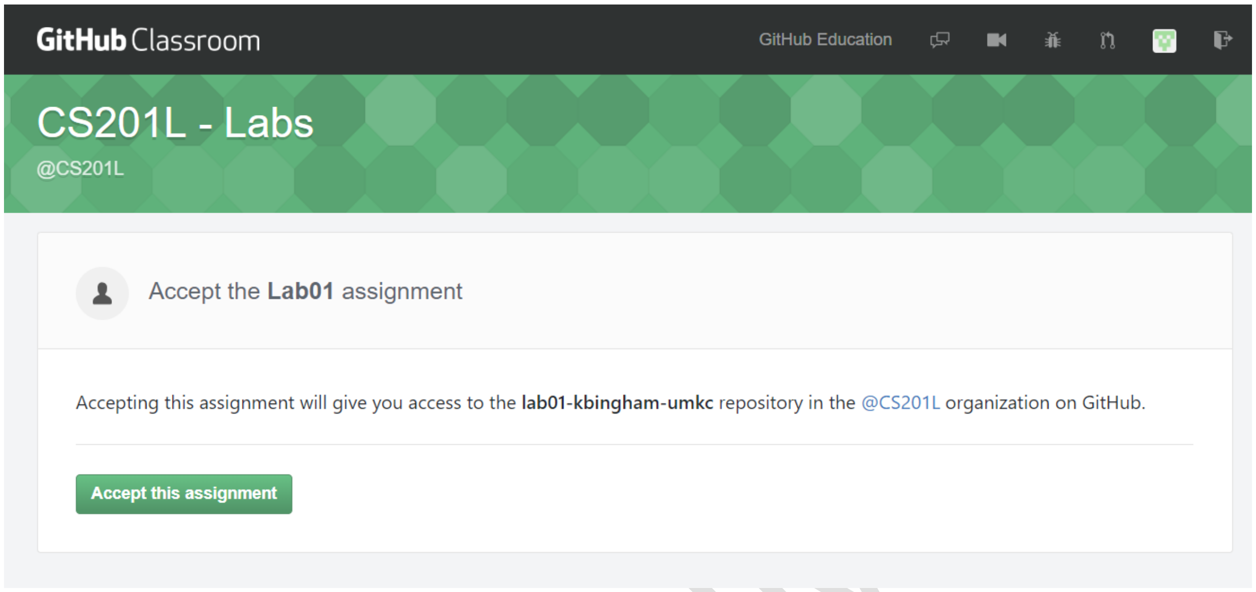
; clone, add, commit and push. You'll also need to be able to work in directories with command line tools. In windows, you'll want to familiarize yourself with bash commands.

Command	Action
cd path	Changes directory to the one given. Can be relative or full path
ls	Lists files in a directory

1. Create an account on http://www.github.com You will need to use your official UMKC email address. It must match the email address for the classroom.
2. [Optional] Once you've created a github account you can request private repositories. This is optional, but you may want to start using github on your own. Having private repositories allows your work to remain invisible to the public. You can visit https://education.github.com/pack to get free private repositories while you are a student.
3. This guide will assume you use git command line on the lab computers. You can download git from https://git-scm.com/

Each week you'll have a new link for the assignment. You'll use the following directions each week with the URL given for the weeks assignment on Canvas

4. After navigating to the URL given for the week click on Accept this assignment



GitHub Classroom

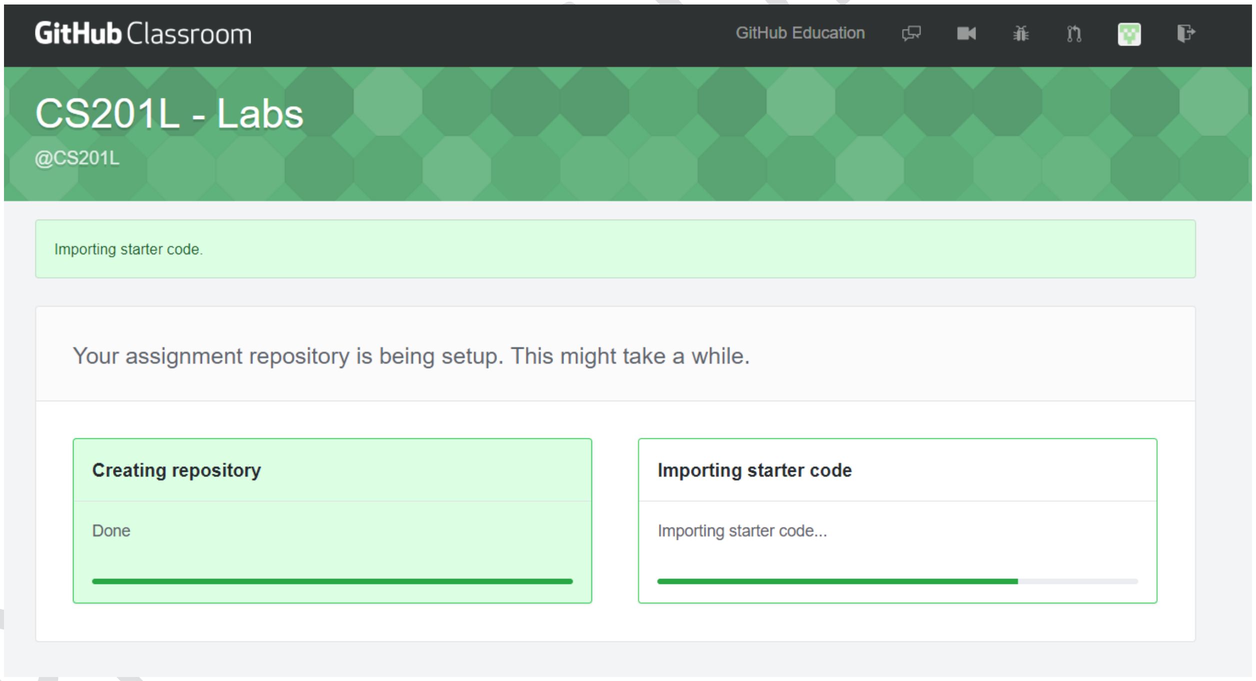
CS201L - Labs

@CS201L

Accept the **Lab01** assignment

Accepting this assignment will give you access to the **lab01-kbingham-umkc** repository in the @CS201L organization on GitHub.

Once you've accepted the assignment, it will begin creating your repository



GitHub Classroom

CS201L - Labs

@CS201L

Importing starter code.

Your assignment repository is being setup. This might take a while.

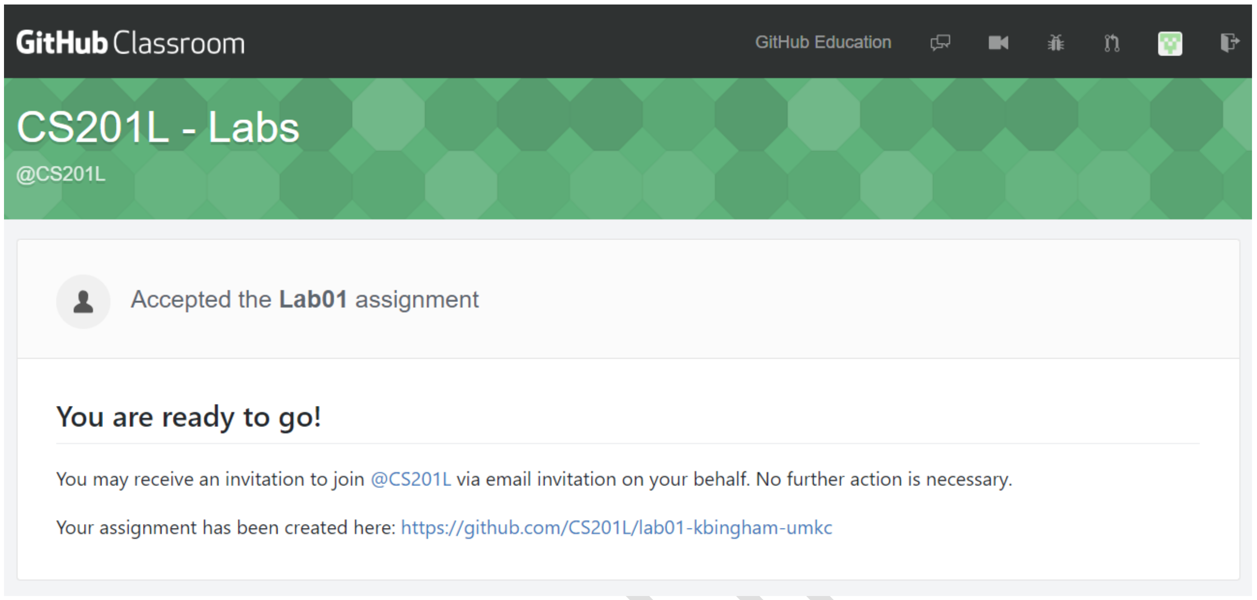
Creating repository

Done

Importing starter code

Importing starter code...

Click on the link for your assignment has been created here..



CS201L - Labs
@CS201L

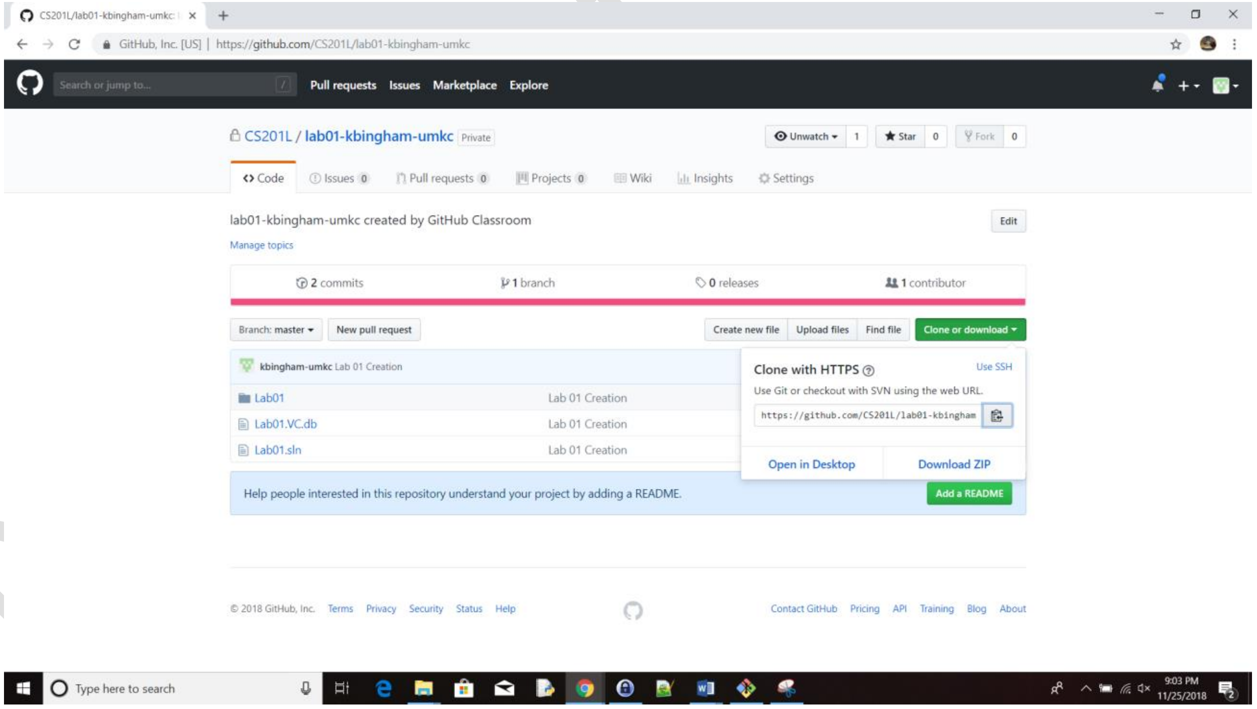
Accepted the **Lab01** assignment

You are ready to go!

You may receive an invitation to join @CS201L via email invitation on your behalf. No further action is necessary.

Your assignment has been created here: https://github.com/CS201L/lab01-kbingham-umkc

Once in your repository click Clone or Download, then copy down the repository address. You'll use this later for cloning the repository.



CS201L / lab01-kbingham-umkc Private

lab01-kbingham-umkc created by GitHub Classroom

2 commits | 1 branch | 0 releases | 1 contributor

kbingham-umkc Lab 01 Creation

Lab01	Lab 01 Creation
Lab01.VC.db	Lab 01 Creation
Lab01.sln	Lab 01 Creation

Clone with HTTPS
Use SSH
Use Git or checkout with SVN using the web URL.

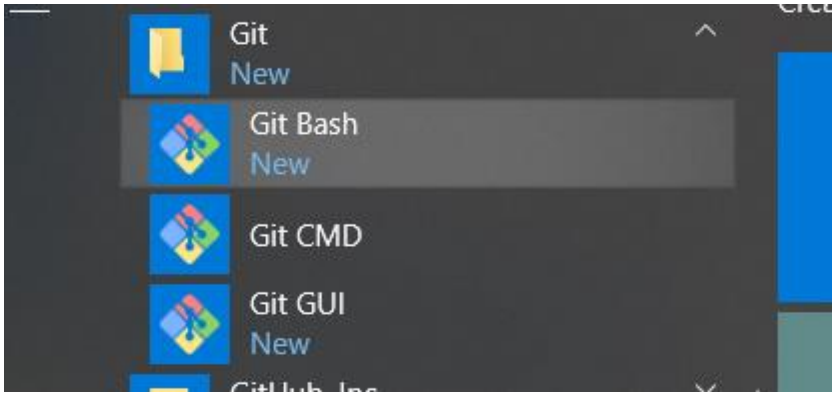
```
https://github.com/CS201L/lab01-kbingham
```

Help people interested in this repository understand your project by adding a README.

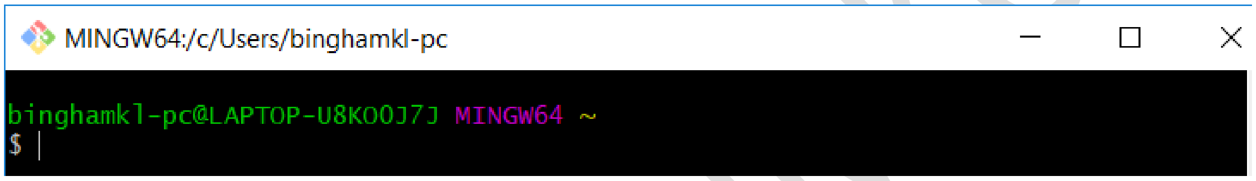
© 2018 GitHub, Inc. Terms Privacy Security Status Help Contact GitHub Pricing API Training Blog About

9:03 PM 11/25/2018

5. Start git BASH. On windows go to start GIT and then click on GIT BASH.
6. Use cd to change to the directory you want to put your work in.

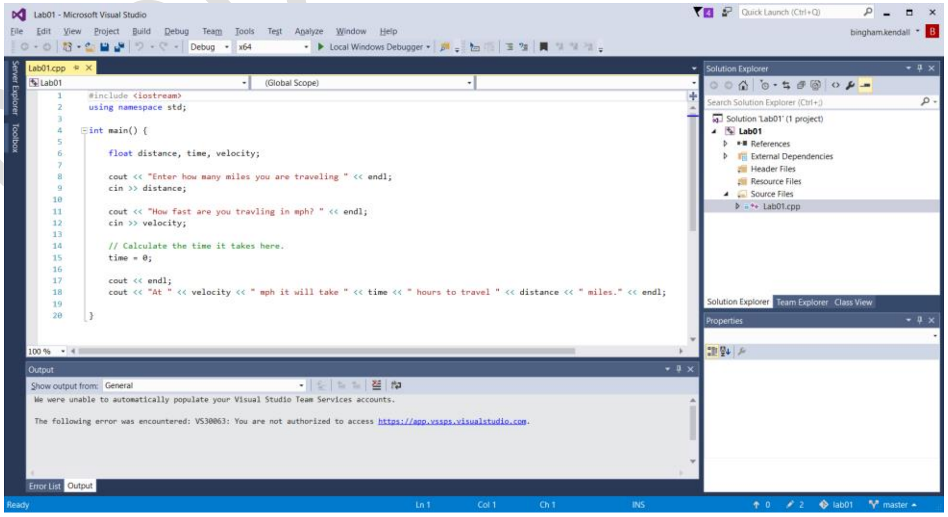


7. The git BASH window is a command line window and looks like the following.



```
MINGW64:/c/Users/binghamkl-pc

binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~
$ |
```

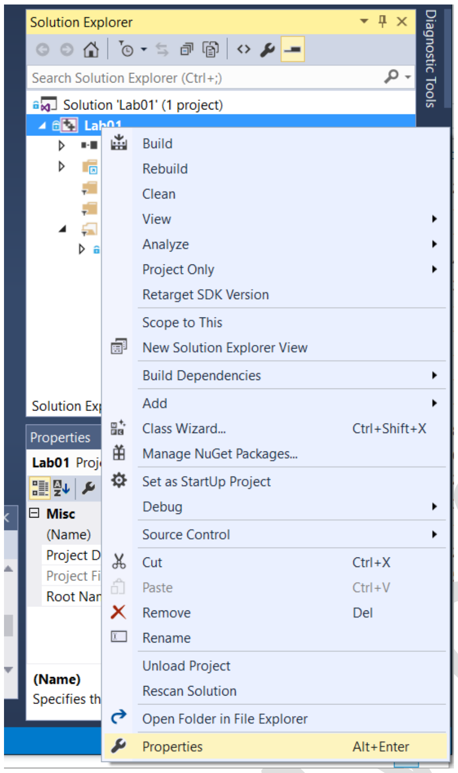
8. You can type commands at the prompt. ls<enter> to view the contents of the current directory. cd to change to a new directory. Go ahead change to a directory where you want to put your lab assignments.
9. Once you are in a directory to put your repository
   Type, instead of <url> enter the repository clone address you got above. Replace xx in labxx with the number for the current week lab.
   `git clone <url> labxx`
10. Once your repository is cloned type the following, replacing labxx with the week you are in.
    cd labxx
11. Now open the .sln file in this directory with Visual Studio 2015, or the C++ compiler of your choice.
12. In this first assignment, change the calculation for time to correctly calculate the time. Run and test your program. Once you've are finished, save all of your work and return to the bash command window. In Solution Explorer on the right open Solution, then Lab01, then Source files. You'll see the Lab01.cpp file, double click on it and it will open in the window.



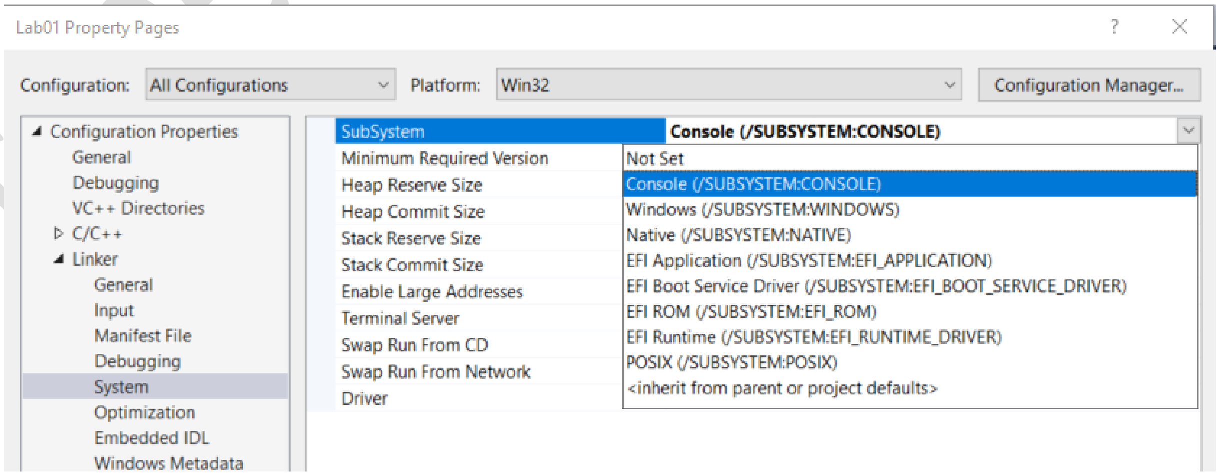
```cpp
#include <iostream>
using namespace std;

int main() {

    float distance, time, velocity;

    cout << "Enter how many miles you are traveling " << endl;
    cin >> distance;

    cout << "How fast are you travling in mph? " << endl;
    cin >> velocity;

    // Calculate the time it takes here.
    time = 0;

    cout << endl;
    cout << "At " << velocity << " mph it will take " << time << " hours to travel " << distance << " miles." << endl;

}
```

```
We were unable to automatically populate your Visual Studio Team Services accounts.

The following error was encountered: VS30063: You are not authorized to access https://app.vssps.visualstudio.com.
```

13. **Keeping the console window open after the program runs.** When you run the program in debug mode Ctrl-F5 you may notice the console window closes before you can see the results of the program. To keep this from happening in Visual Studio follow the next steps.
    a. Right click on Lab01 project as shown below. Go down and select properties at the bottom of the menu.



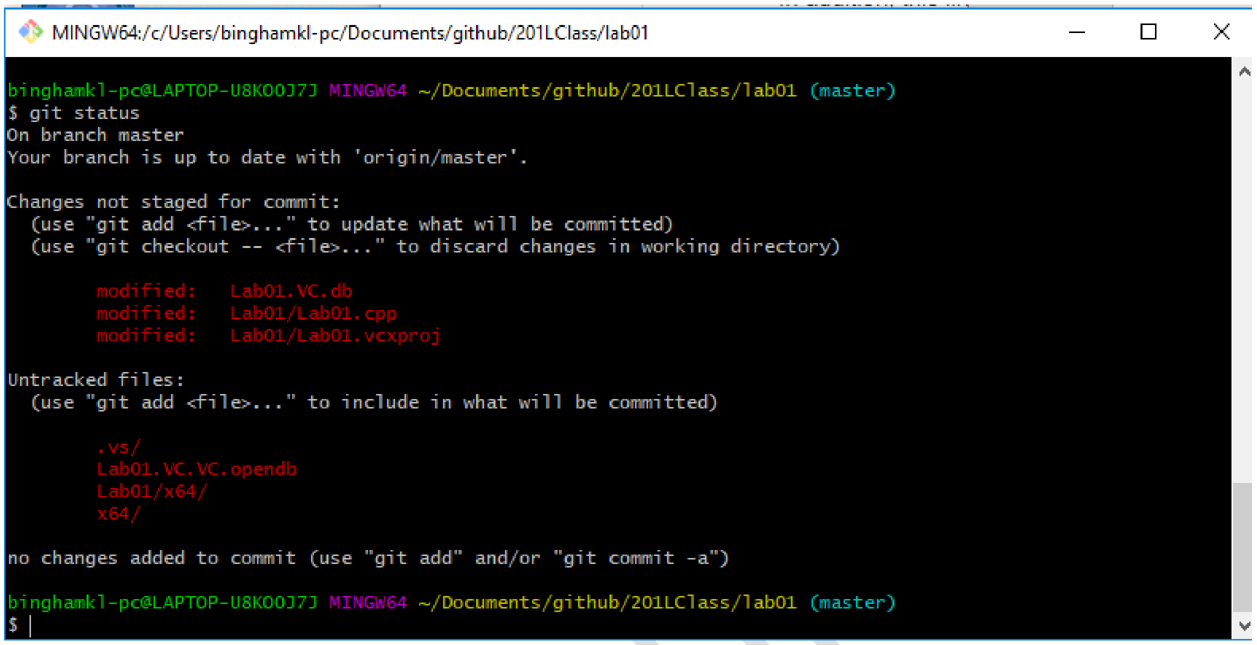
    b. With the property window open, select Linker, then System on the left. Open the SubSystem drop down box and select console. Hit ok, and your debug console window should remain open after your code runs. Make sure the configuration is set to All Configurations.



Configuration: All Configurations
Platform: Win32
SubSystem: Console (/SUBSYSTEM:CONSOLE)

14. In git bash shell type the following line to see the status of the local repository.

```
git status
```


MINGW64:/c/Users/binghamkl-pc/Documents/github/201LClass/lab01

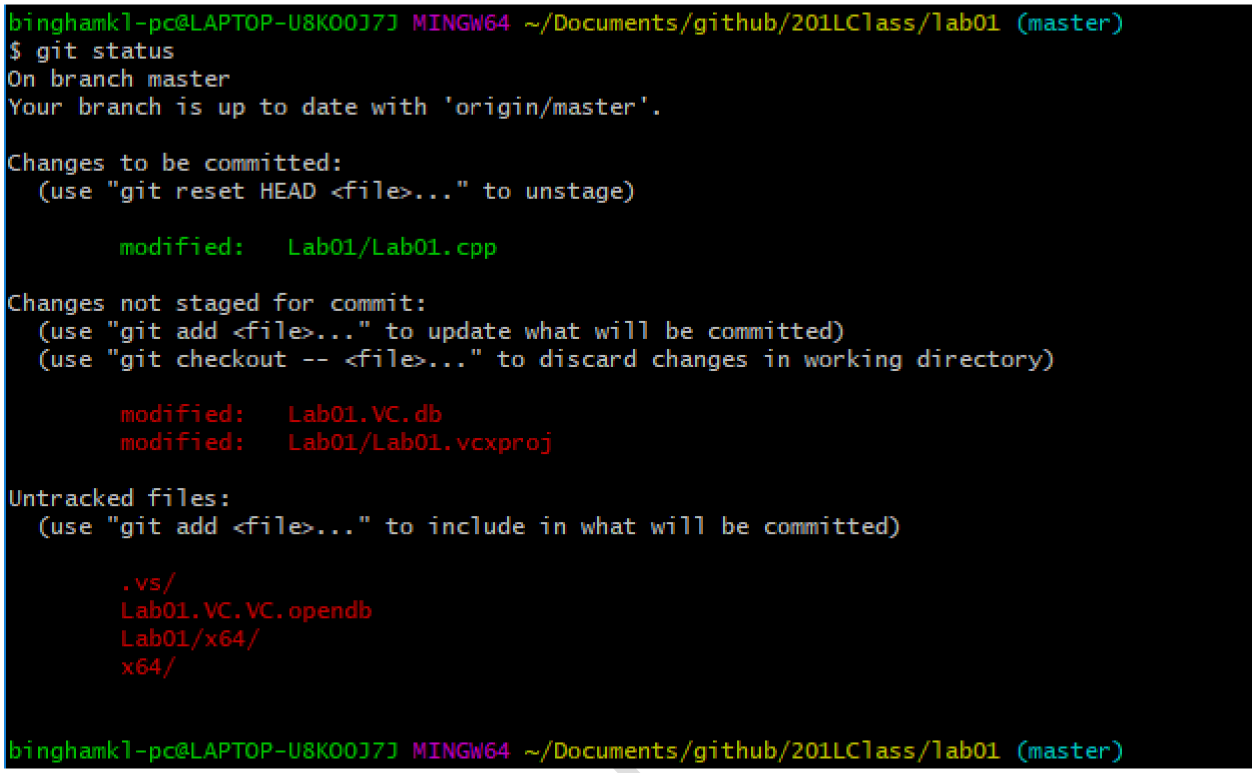
```
binghamkl-pc@LAPTOP-U8KOOJ7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Lab01.VC.db
        modified:   Lab01/Lab01.cpp
        modified:   Lab01/Lab01.vcxproj

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

        .vs/
        Lab01.VC.VC.opendb
        Lab01/x64/
        x64/

no changes added to commit (use "git add" and/or "git commit -a")

binghamkl-pc@LAPTOP-U8KOOJ7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
$ |
```

You'll notice that we have changes staged, and untracked files. Untracked files are files that the source control isn't keeping track of. If you add a new file that needs to be tracked, then you can use the git add command to add the file.

The files we've changed show up in changes not staged. We use the git add command to stage these.

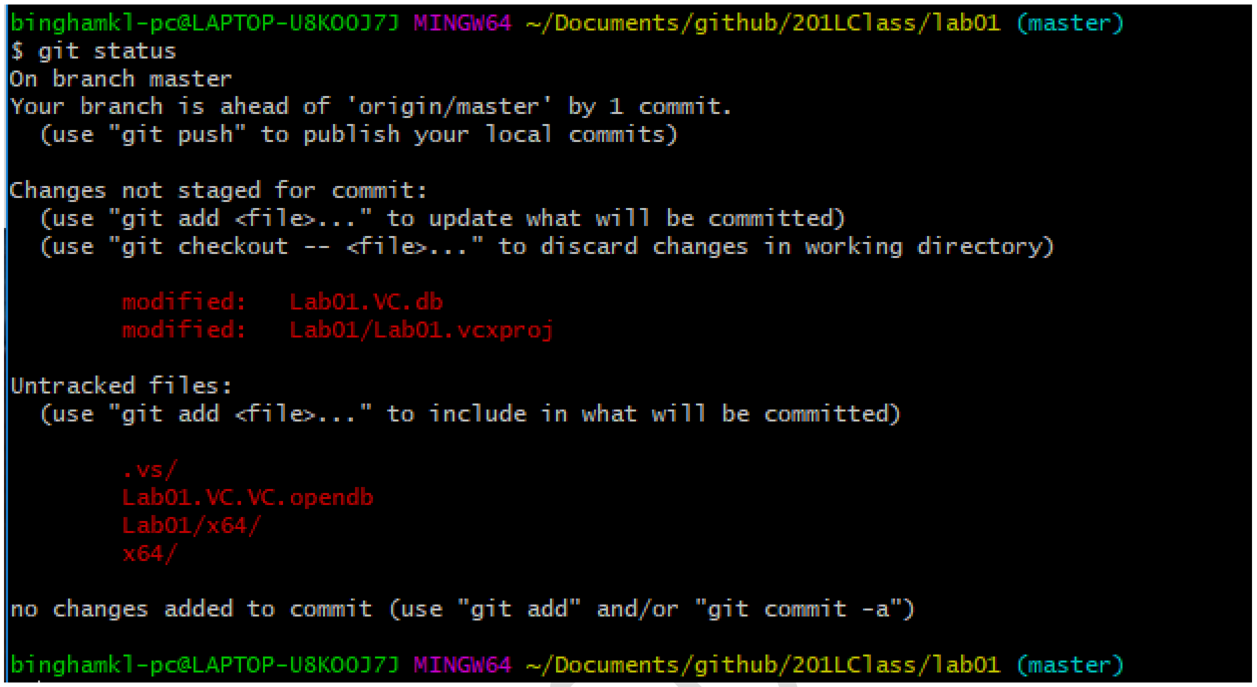
15. In the git bash we first need to add all the files that have been changed
`git add Lab01/Lab01.cpp`

16. Run git status again and you should see the changes that need to be committed.


```
binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (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)

        modified:   Lab01/Lab01.cpp

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Lab01.VC.db
        modified:   Lab01/Lab01.vcxproj

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

        .vs/
        Lab01.VC.VC.opendb
        Lab01/x64/
        x64/


binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
```

Notice that we now have changes that need to be committed. Lab01.cpp is in the staged area, but is not committed to our local repository.

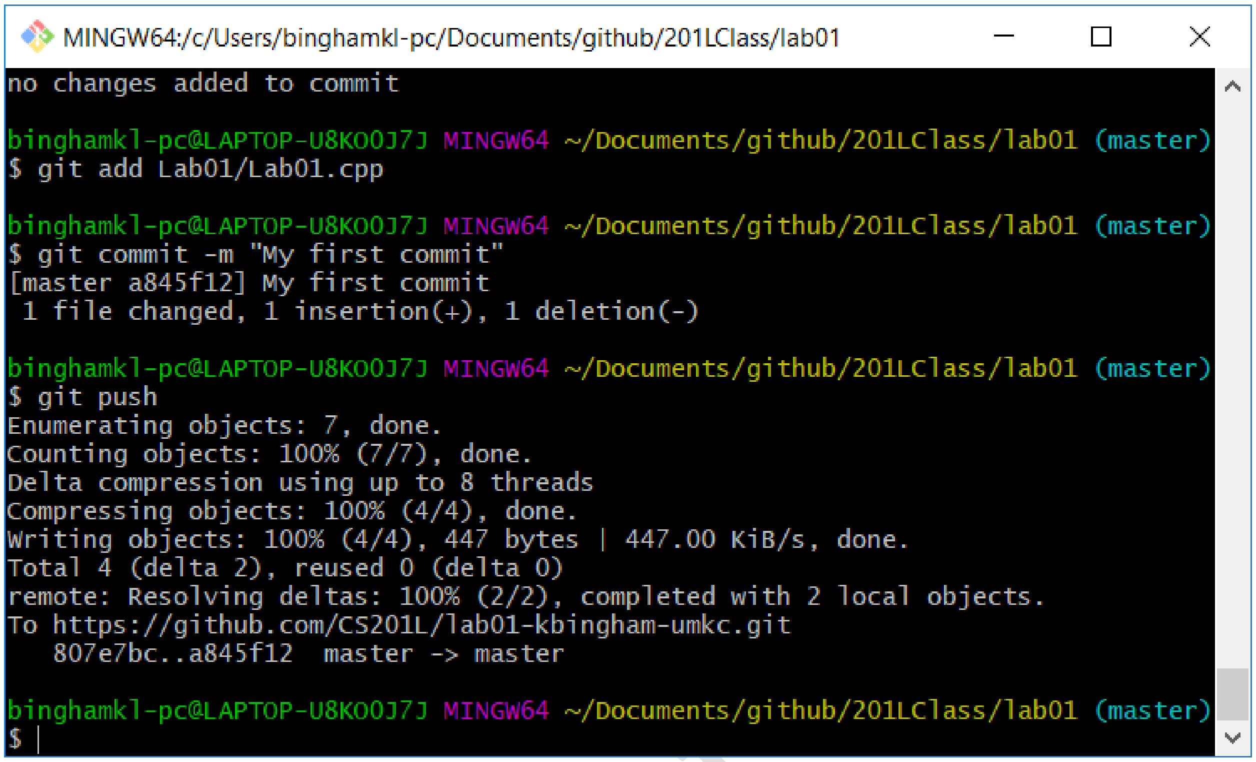
17. Then we need to commit those changes to our local repository. You can use whatever message you want in between the “” It should be something that is descriptive to what you have just committed.
    `git commit -m "Push the new change"`
18. Run git status again. You should see something like the following.



```
binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   Lab01.VC.db
        modified:   Lab01/Lab01.vcxproj

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

        .vs/
        Lab01.VC.VC.opendb
        Lab01/x64/
        x64/

no changes added to commit (use "git add" and/or "git commit -a")

binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
```

Notice that our branch is ahead of the master remote repository. Our local repository is up to date, but the remote one has not been updated yet.

19. The only thing left is to push the solution to github repository.
    `git push`

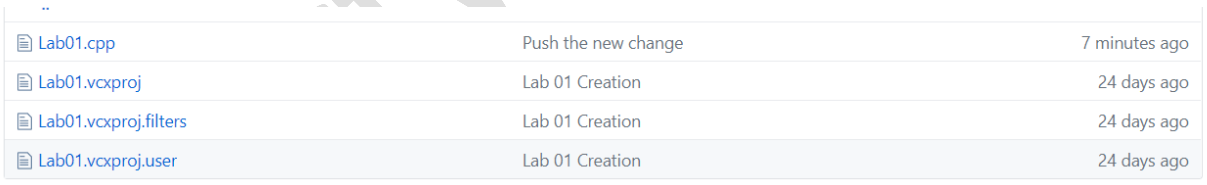


MINGW64:/c/Users/binghamkl-pc/Documents/github/201LClass/lab01

```
no changes added to commit

binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
$ git add Lab01/Lab01.cpp

binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
$ git commit -m "My first commit"
[master a845f12] My first commit
 1 file changed, 1 insertion(+), 1 deletion(-)

binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
$ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 447 bytes | 447.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/CS201L/lab01-kbingham-umkc.git
   807e7bc..a845f12  master -> master

binghamkl-pc@LAPTOP-U8KO0J7J MINGW64 ~/Documents/github/201LClass/lab01 (master)
$ |
```

20. Use git status and you'll notice that you are no longer ahead of the origin/master repository

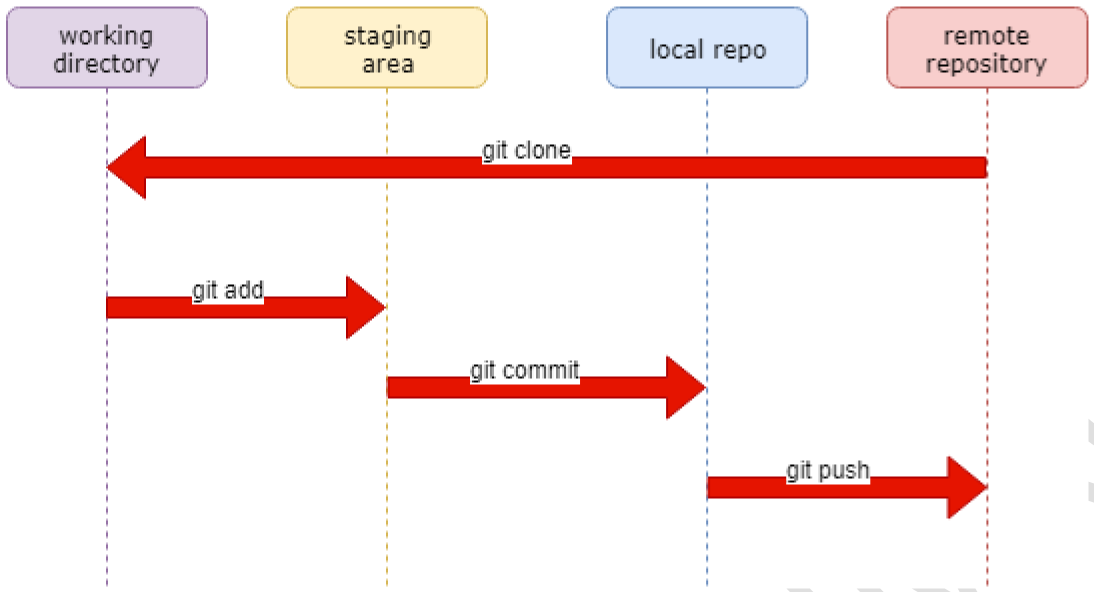
21. Go back to the github link you got for your repository and refresh it, you should be able to view your changes and see that your work has been pushed to github



Lab01.cpp	Push the new change	7 minutes ago
Lab01.vcxproj	Lab 01 Creation	24 days ago
Lab01.vcxproj.filters	Lab 01 Creation	24 days ago
Lab01.vcxproj.user	Lab 01 Creation	24 days ago

22. You can use git add, git commit and git push as many times you need while you are working on a project.

The general git processes


working directory
staging area
local repo
remote repository
git clone
git add
git commit
git push