Syed Arham Gillani (153165)

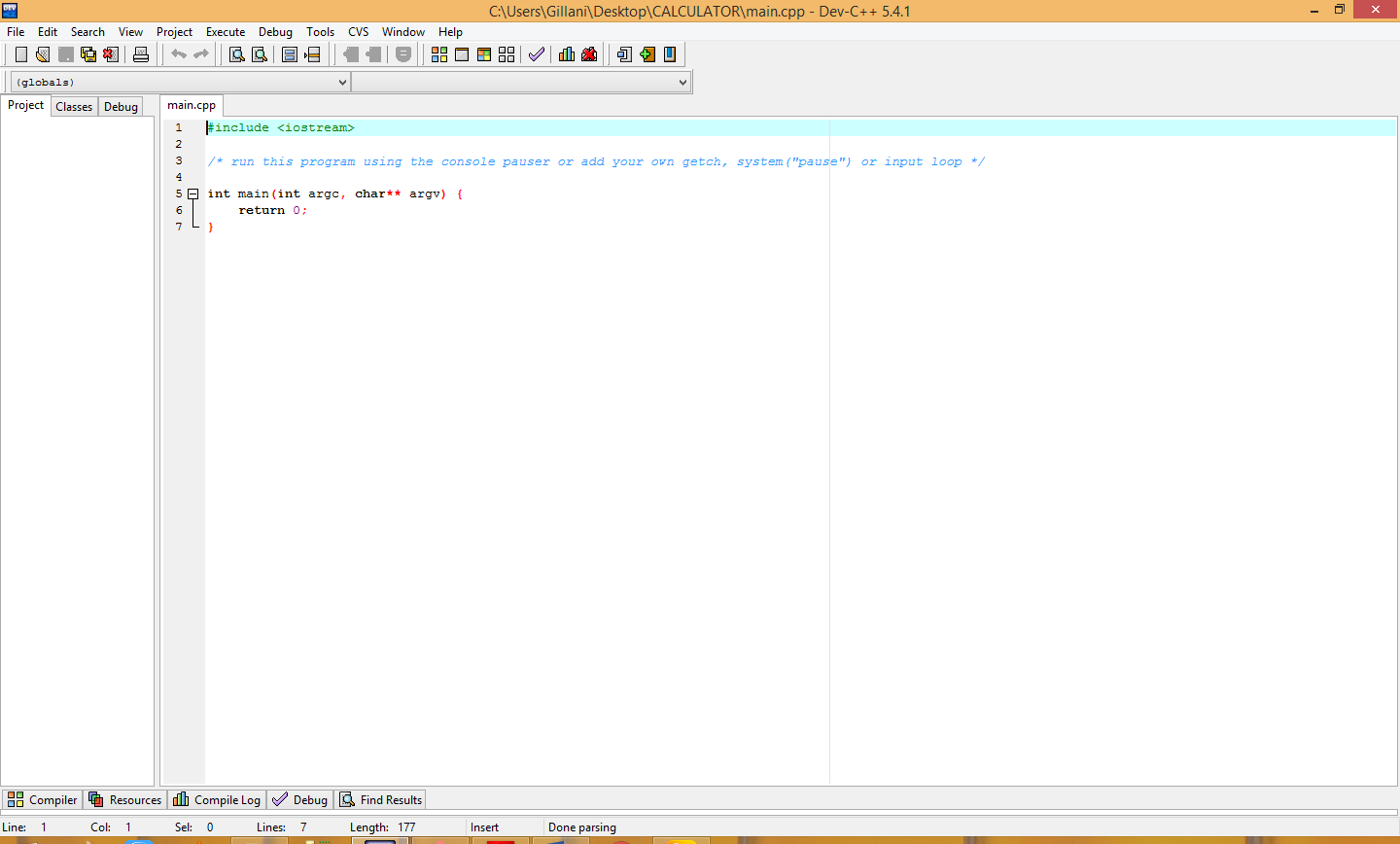
https://github.com/beingGillani/Calculator

Software Engineering Assignment # 1

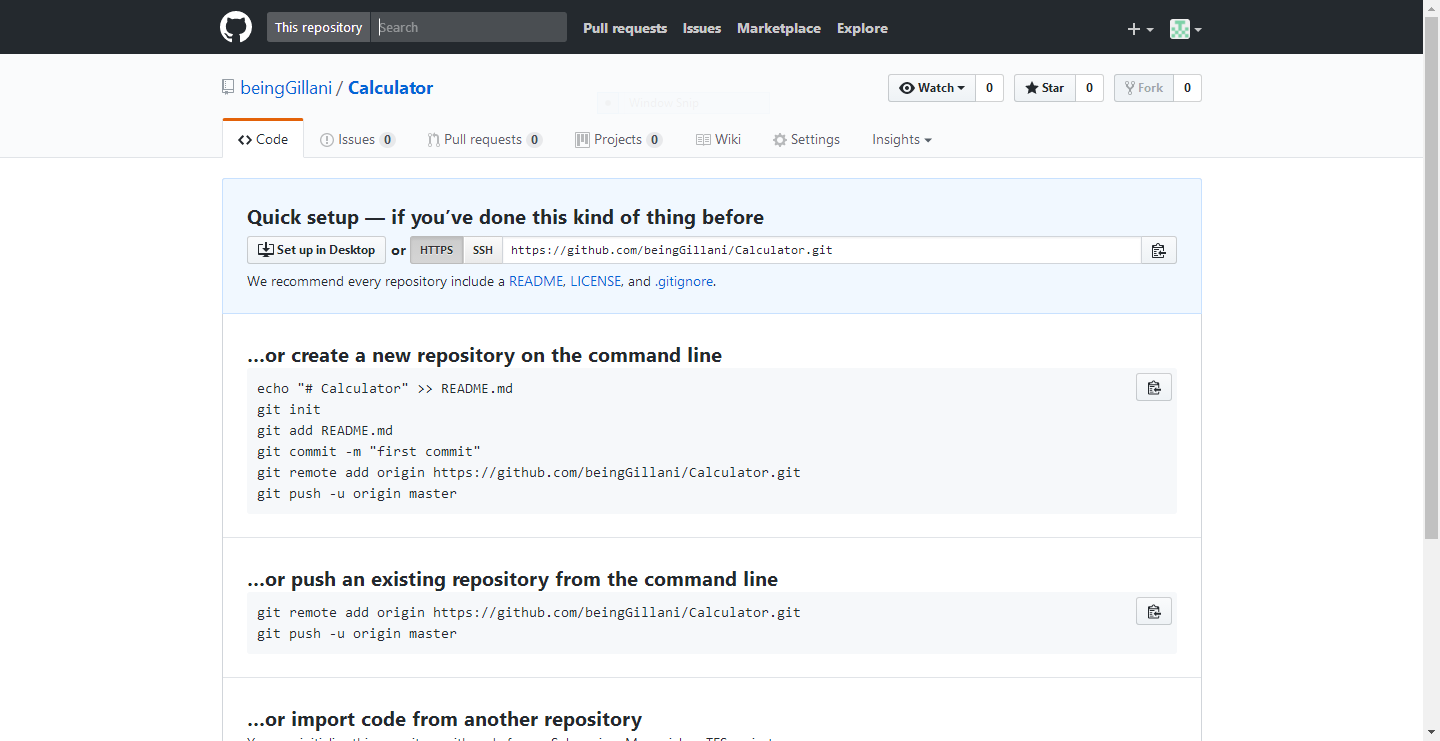
**Exercise 0 – Putting a C++ Project on GitHub**

This exercise will show you how to put a local C++ Project on GitHub.

Step 1. Create an A C++ PROJECT IN DEV OR CODEBLOCKS. I called mine CALCULATOR.



Step 2. Create a repository on GitHub of the same name

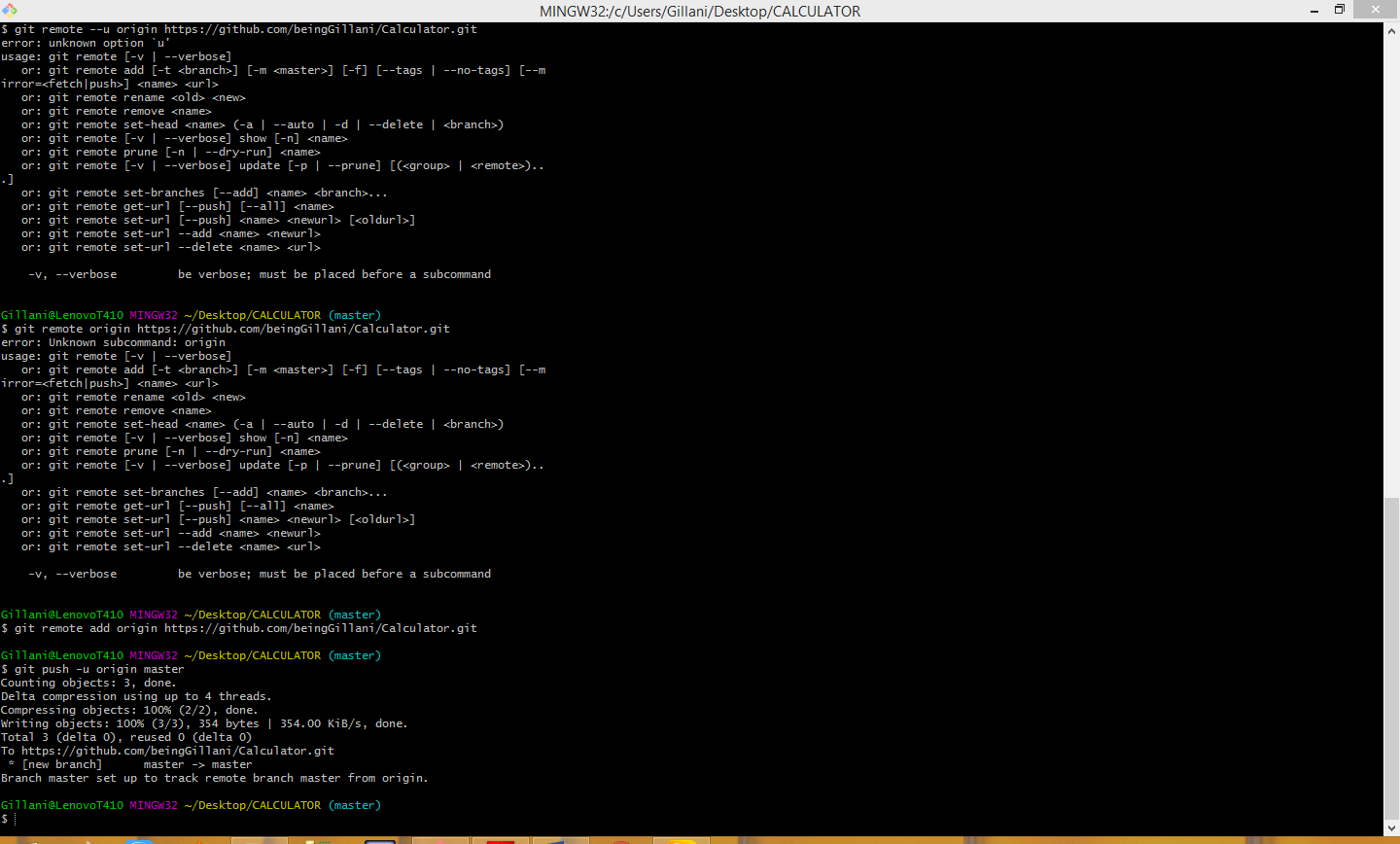


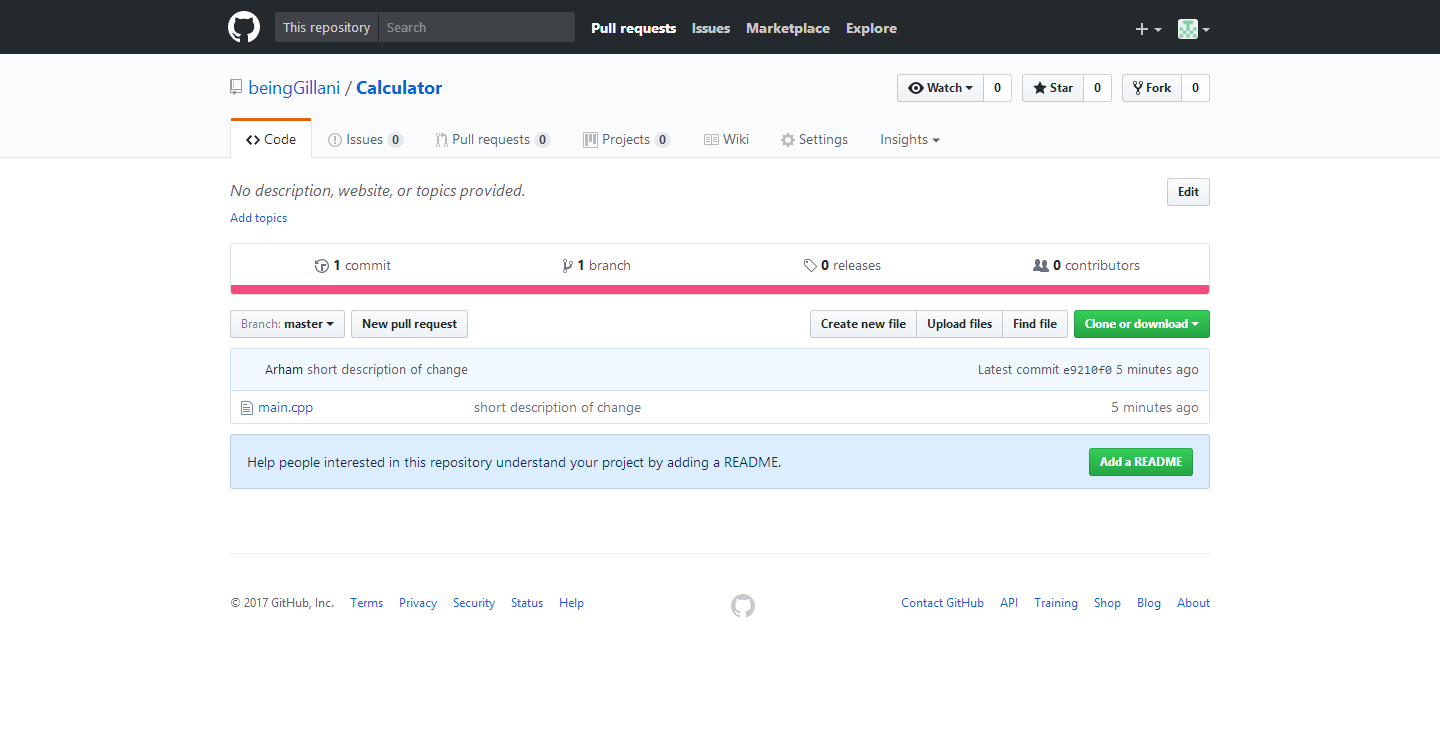
After a change to your project:

$ git add .

$ git commit -m "short description of change"

$ git push -u origin master





\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Exercise 1 – Creating and using a local repository**

Show your current directory (pwd = print working directory):

$ pwd

Create a directory:

$ mkdir gitpractice

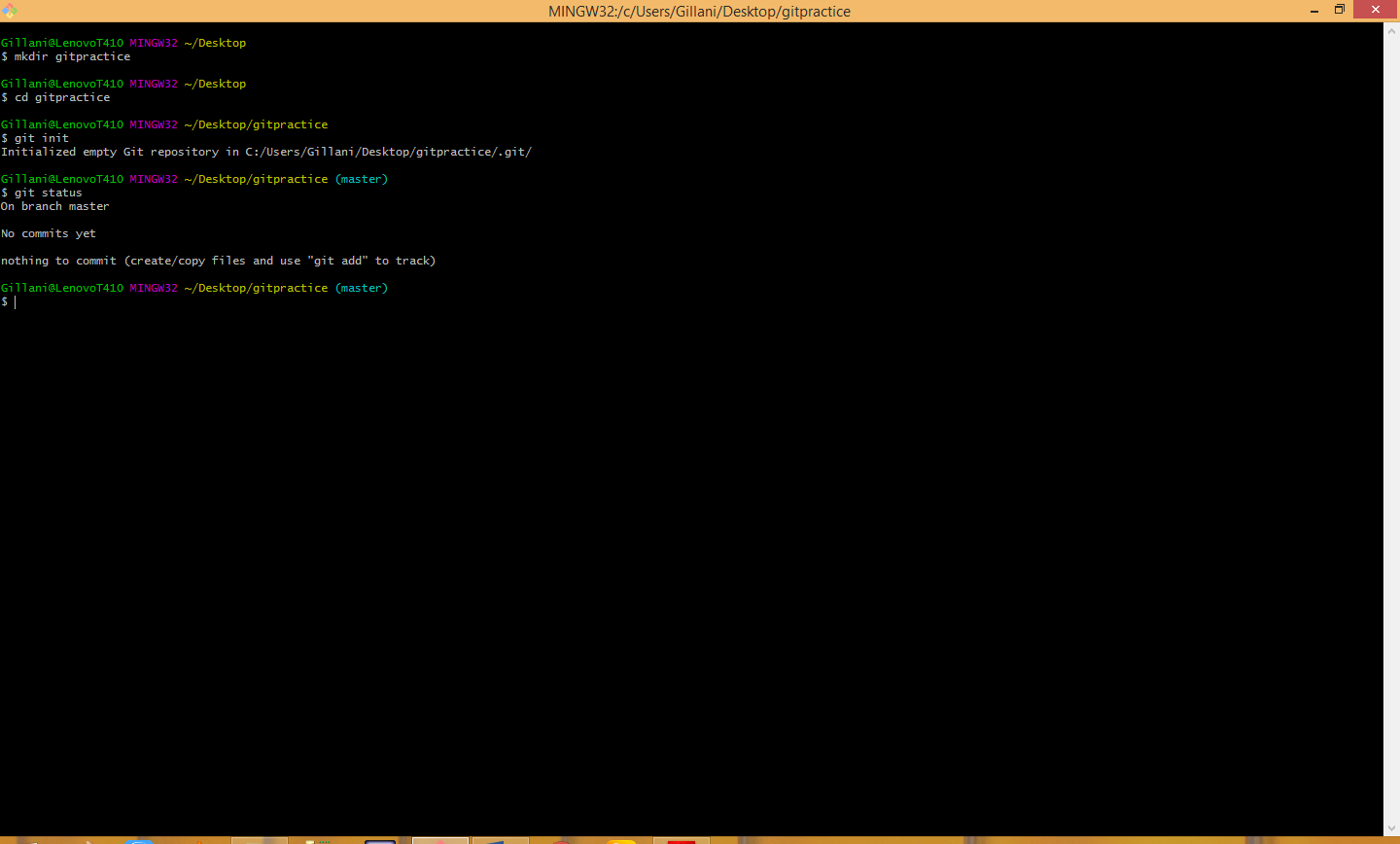
$cd gitpractice

Start tracking files and subfolders with git:

$ git init

Run git status to see current status of the repository:

$ git status



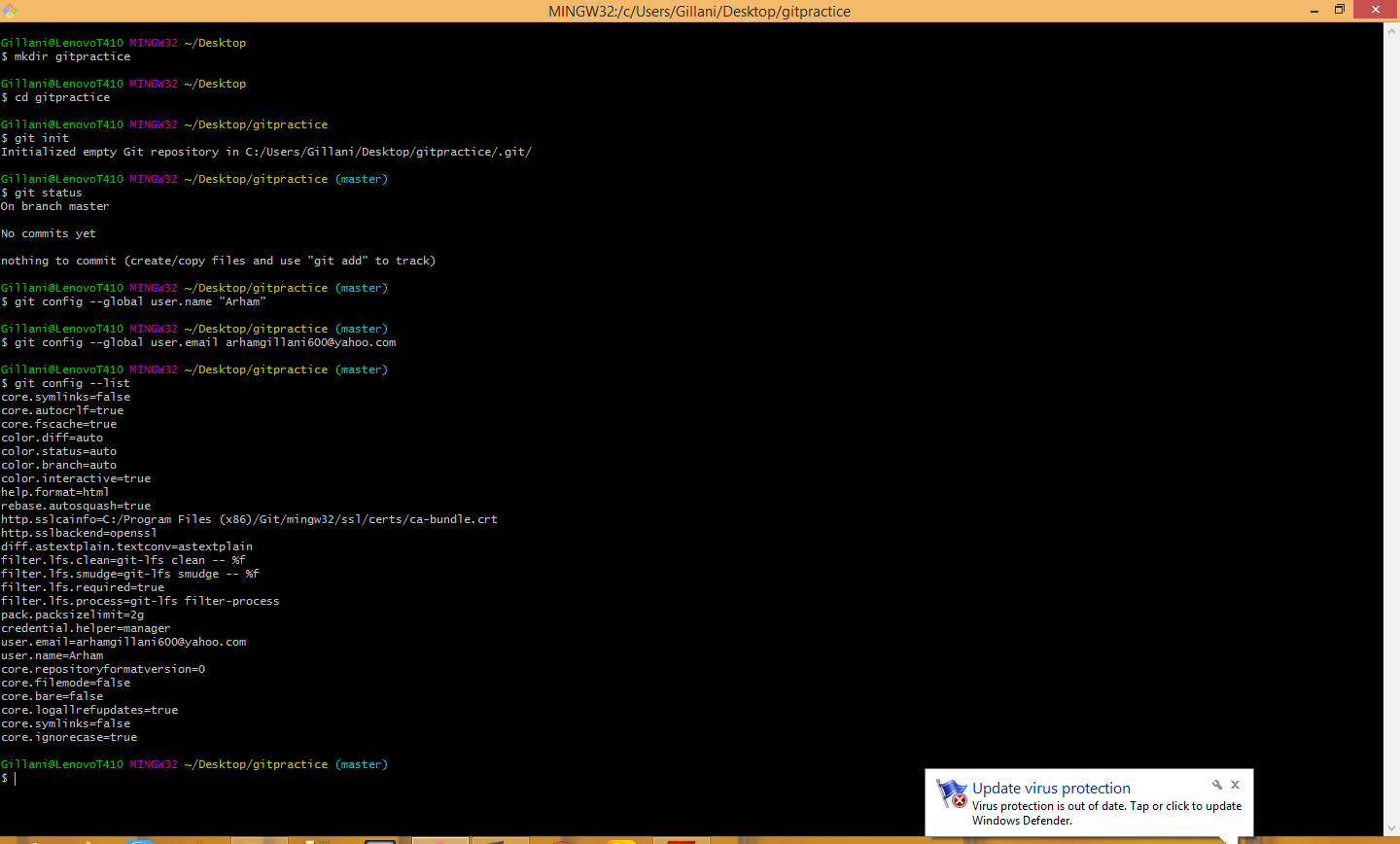
To set your identity, enter:

$ git config --global user.name "Your Name"

$ git config --global user.email you@example.com

Check values you just set:

$ git config --list



You may also want to change the editor. To change the editor to notepad, enter:

***Hopefully before this you have set username and Email through git config command***

$ git config --global core.editor notepad

Add a new file to the directory. For example, use notepad to create a file:

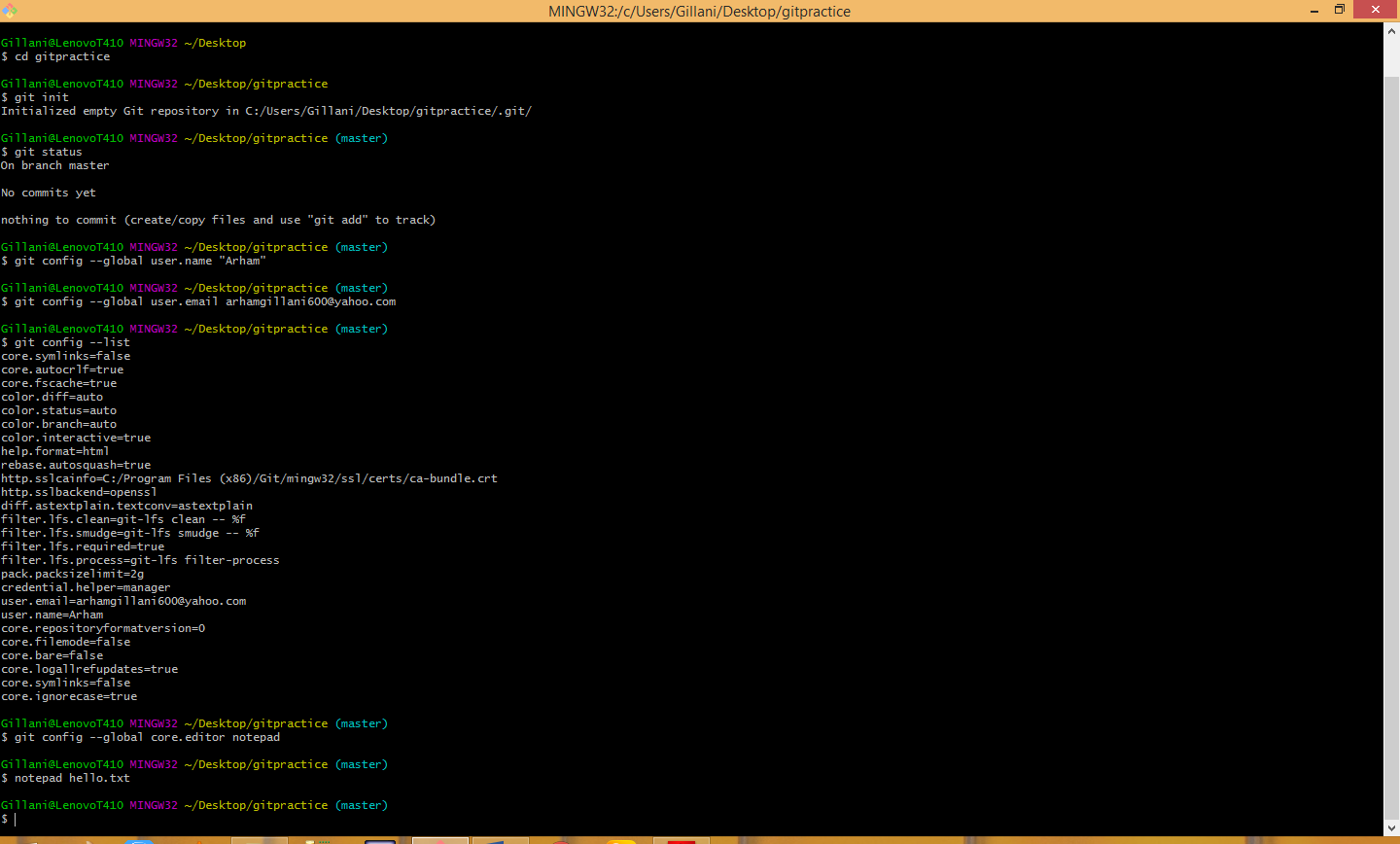
**$ notepad hello.txt**

**And add the contents:**

// **Hello world void main() {**

**print “Hello World”;**

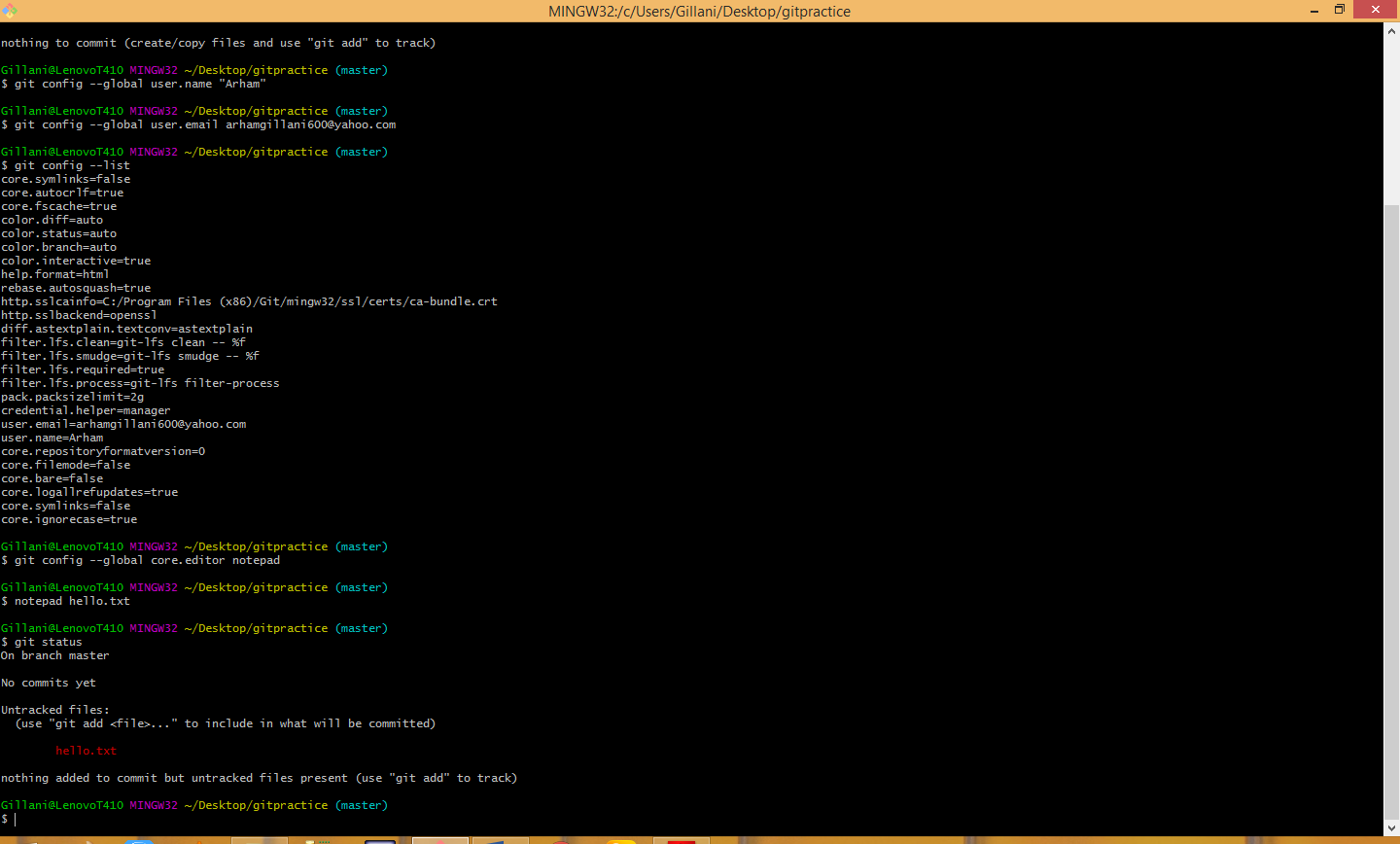
**}**



Run git status:

$ git status

You should see the file as “untracked”.



Add the file with:

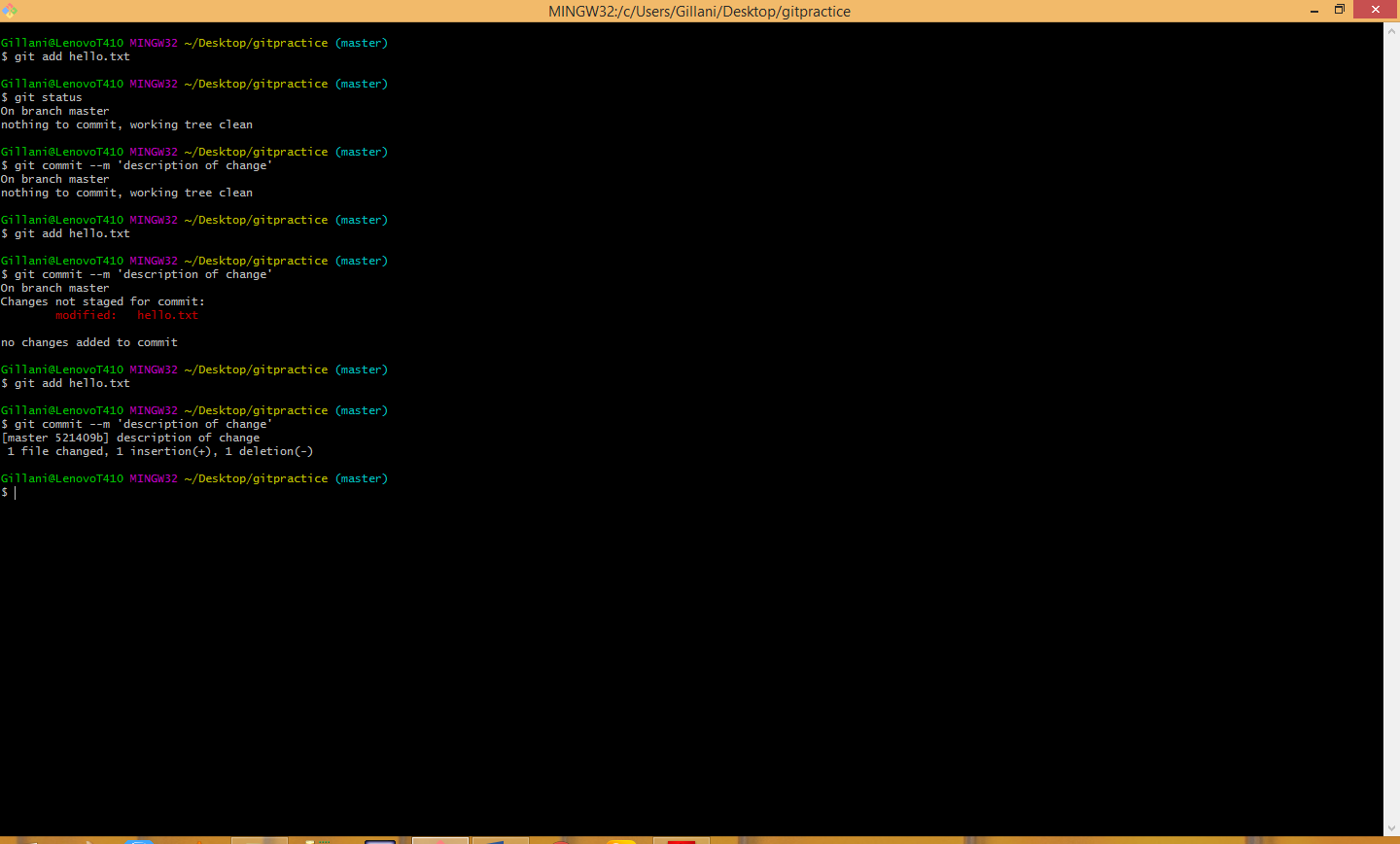
**$ git add hello.txt**

Run git status to verify the file was staged:

**$ git status**

To commit all changes that have been added, enter:

**$ git commit –m ‘description of change’**



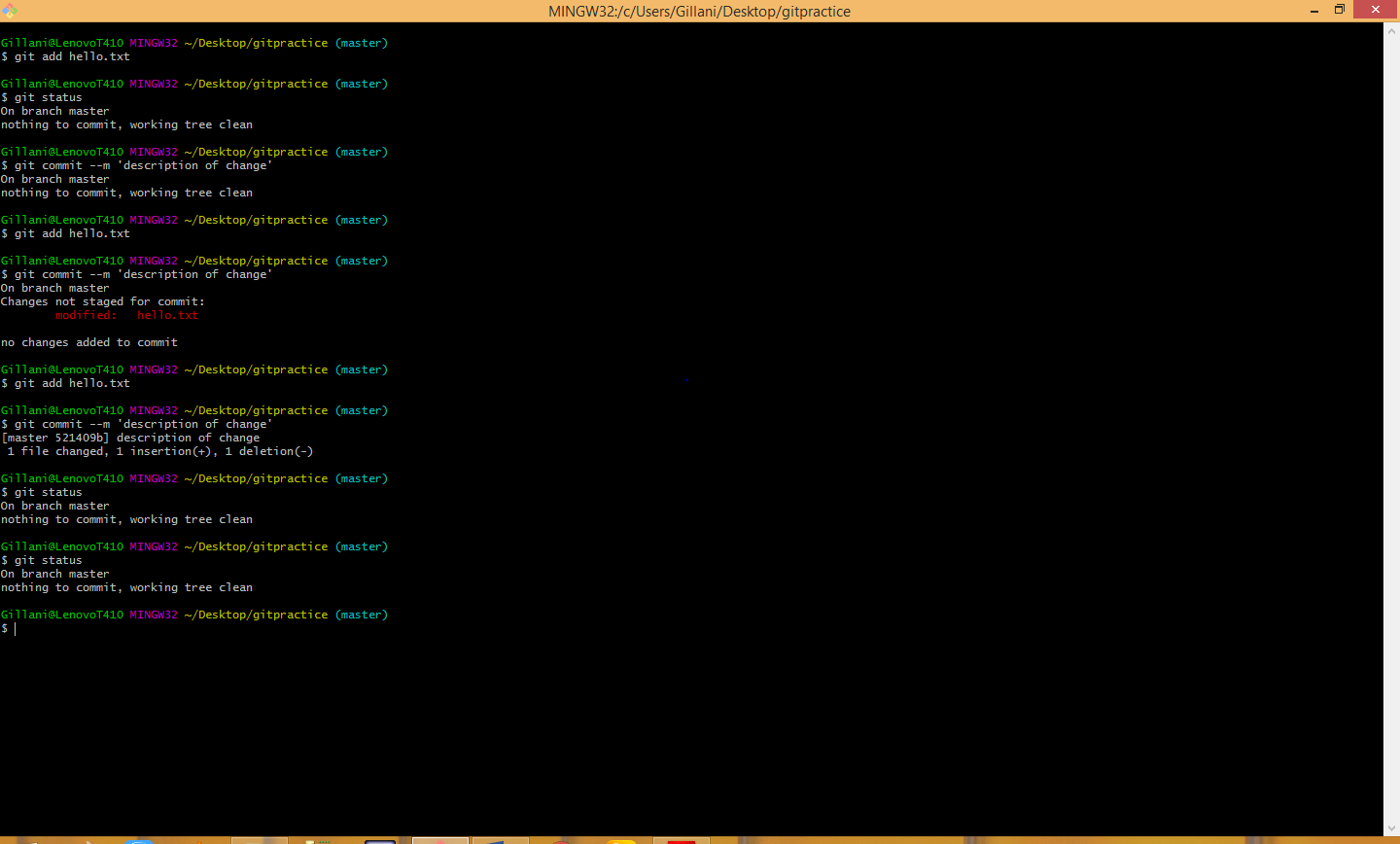
**$ git status**

Make a change to the file using notepad or other editor.

Run git status:

**$ git status**

You should see the status of the file as changed but not staged.

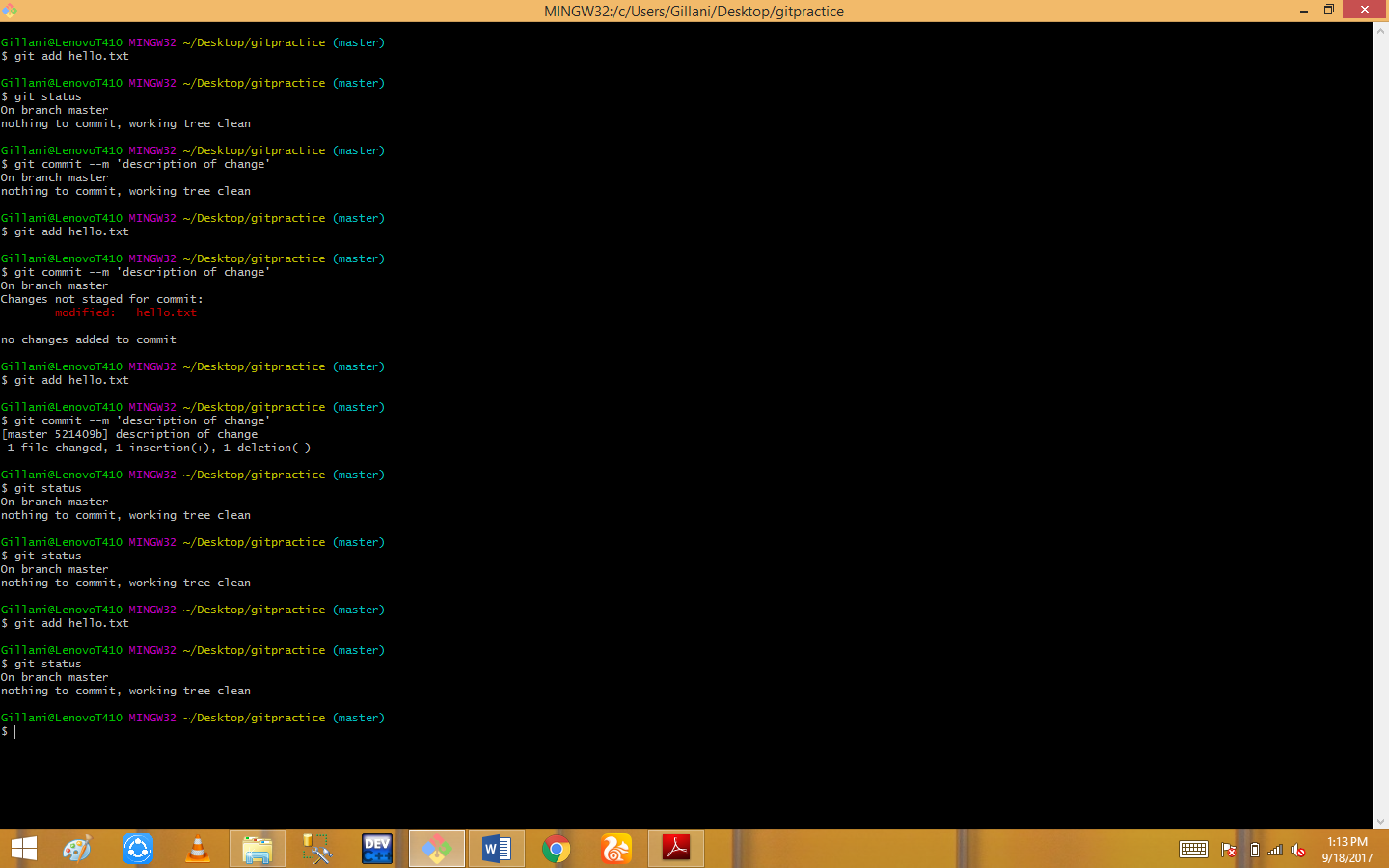


***Stage it with:***

$ git add hello.txt

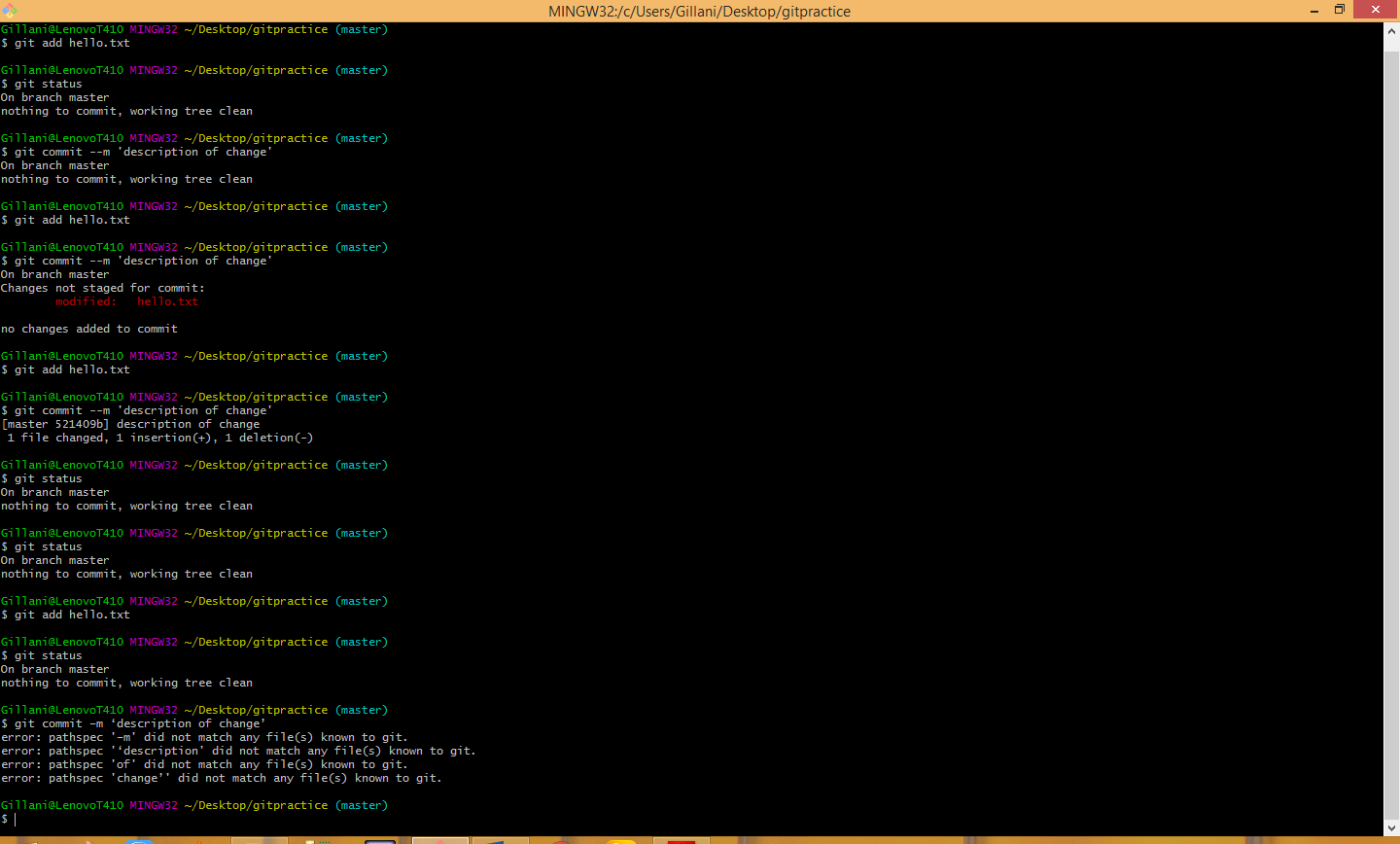
**Run git status:**

$ git status



Now, commit the changes with:

**$ git commit –m ‘description of change’**



**$ git commit**

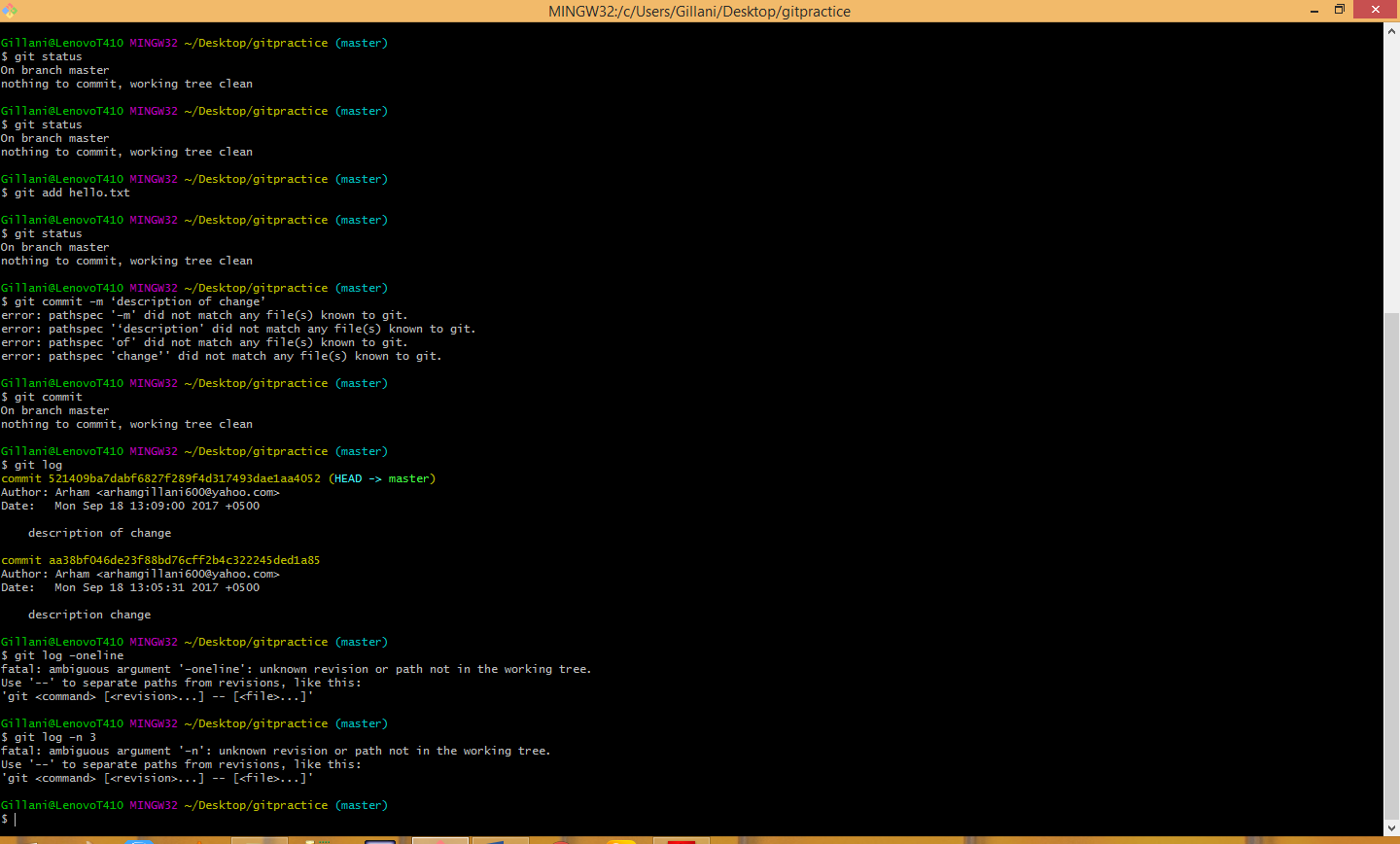
To see a history of changes:

**$ git log**

To see an abbreviated history:

**$ git log –oneline**

**$ git log –n 3**

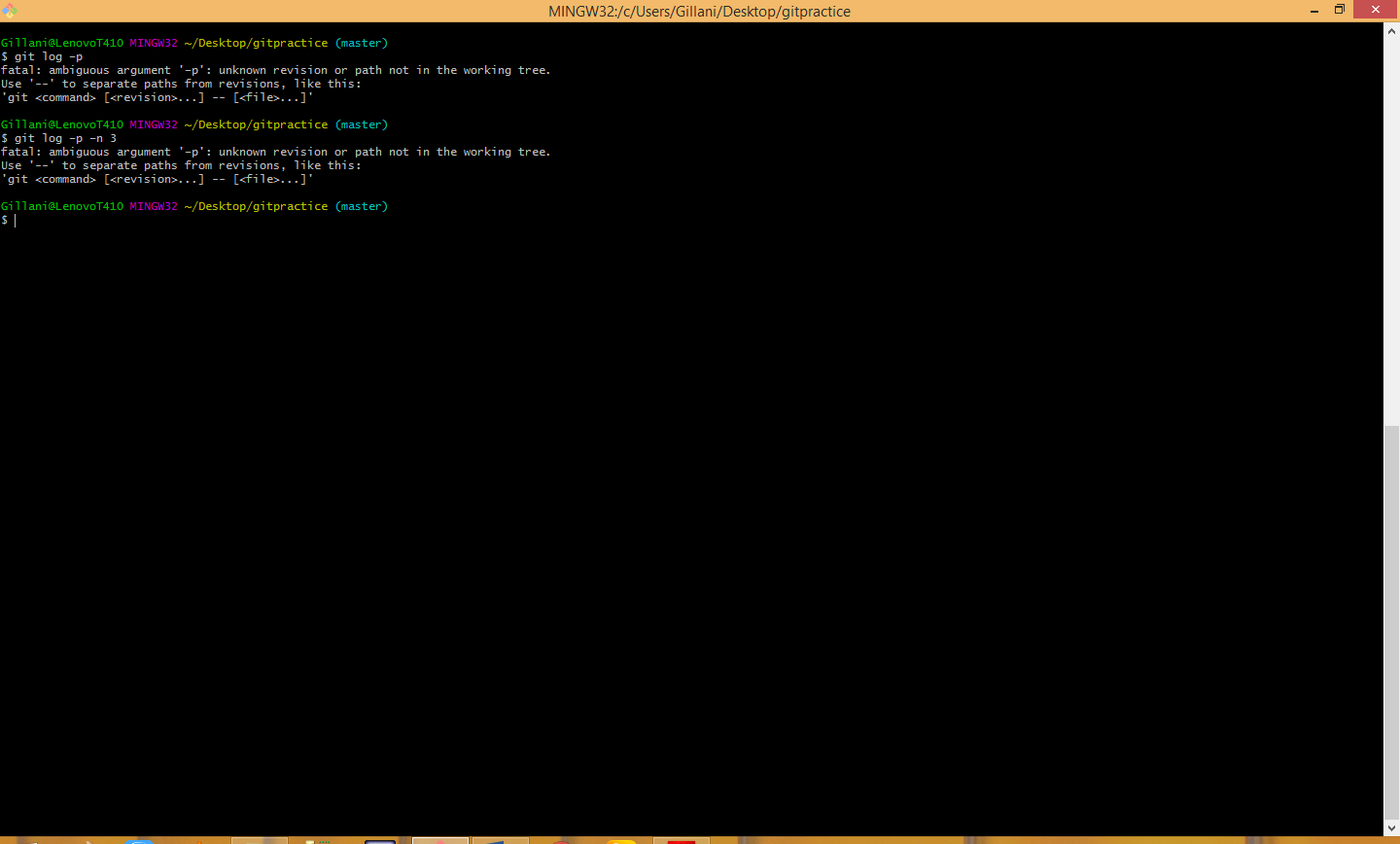


To show the differences introduced in each commit, use the –p option:

**$ git log –p**

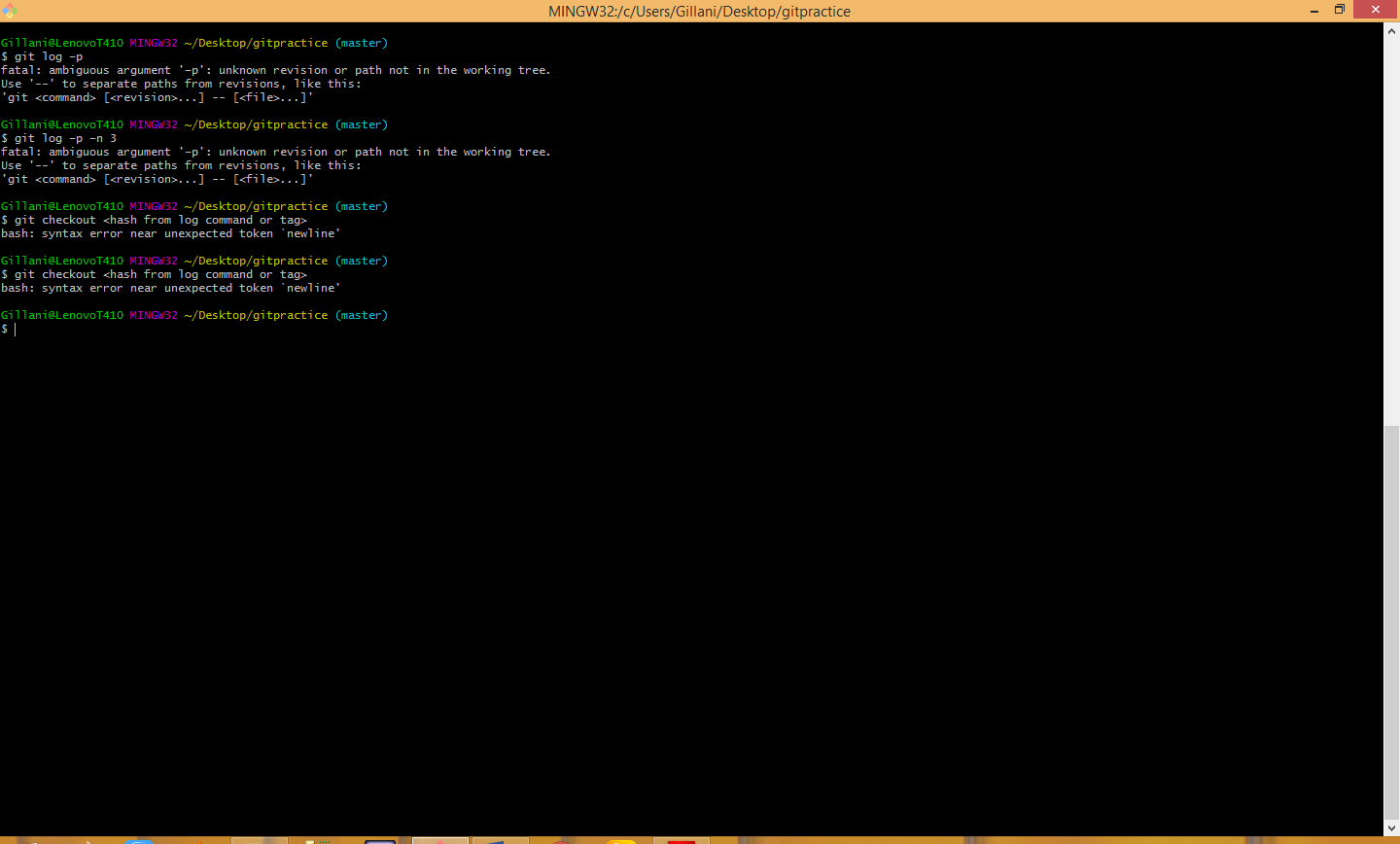
You can of course, combine the two:

**$ git log –p –n 3**



To get back to a previous state:

**$ git checkout <hash from log command or tag>**

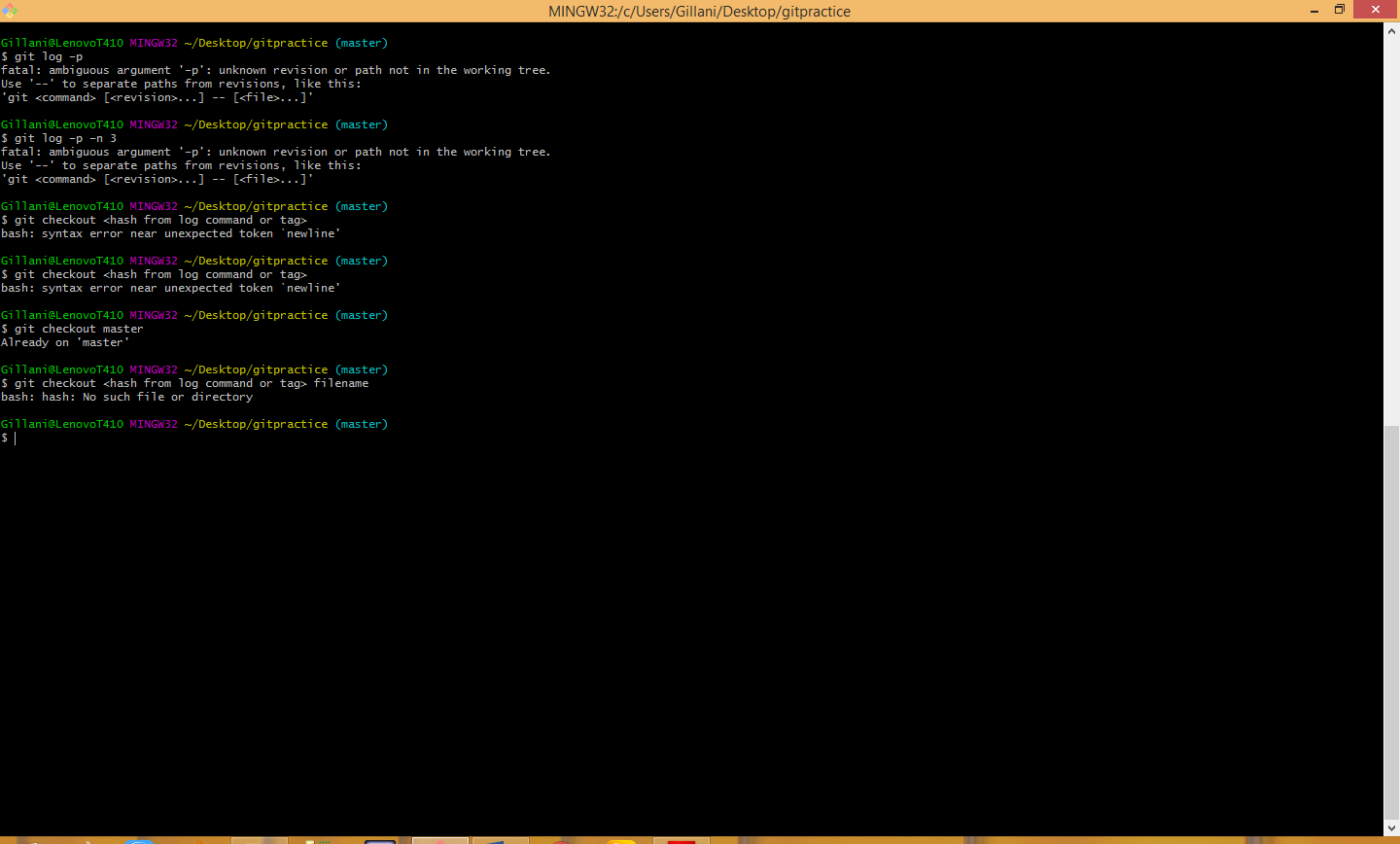


To get back to the latest:

**$ git checkout master**

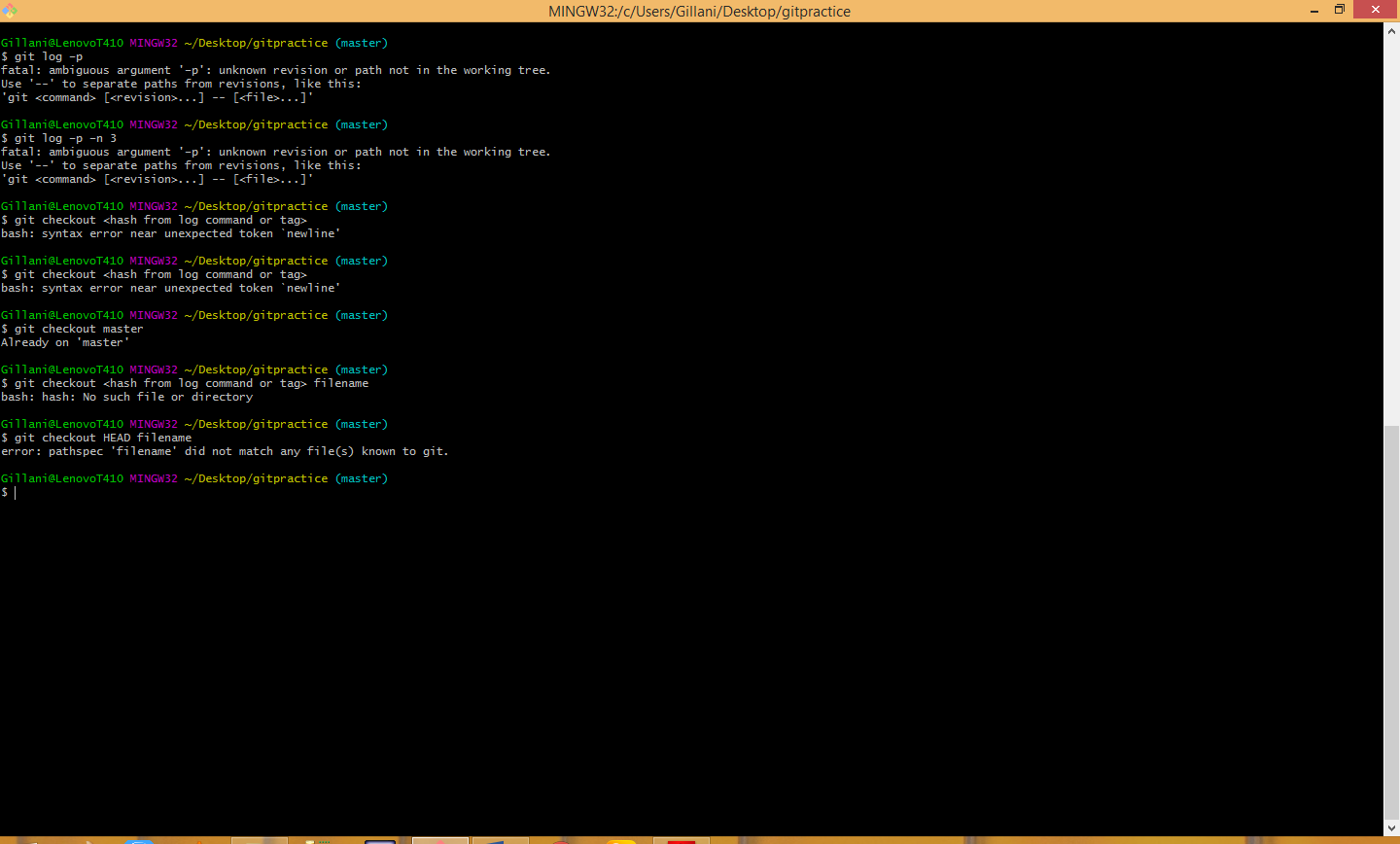
You can also checkout a single file from a previous commit:

$ git checkout <hash from log command or tag> filename



To get back to the latest version of the file, enter:

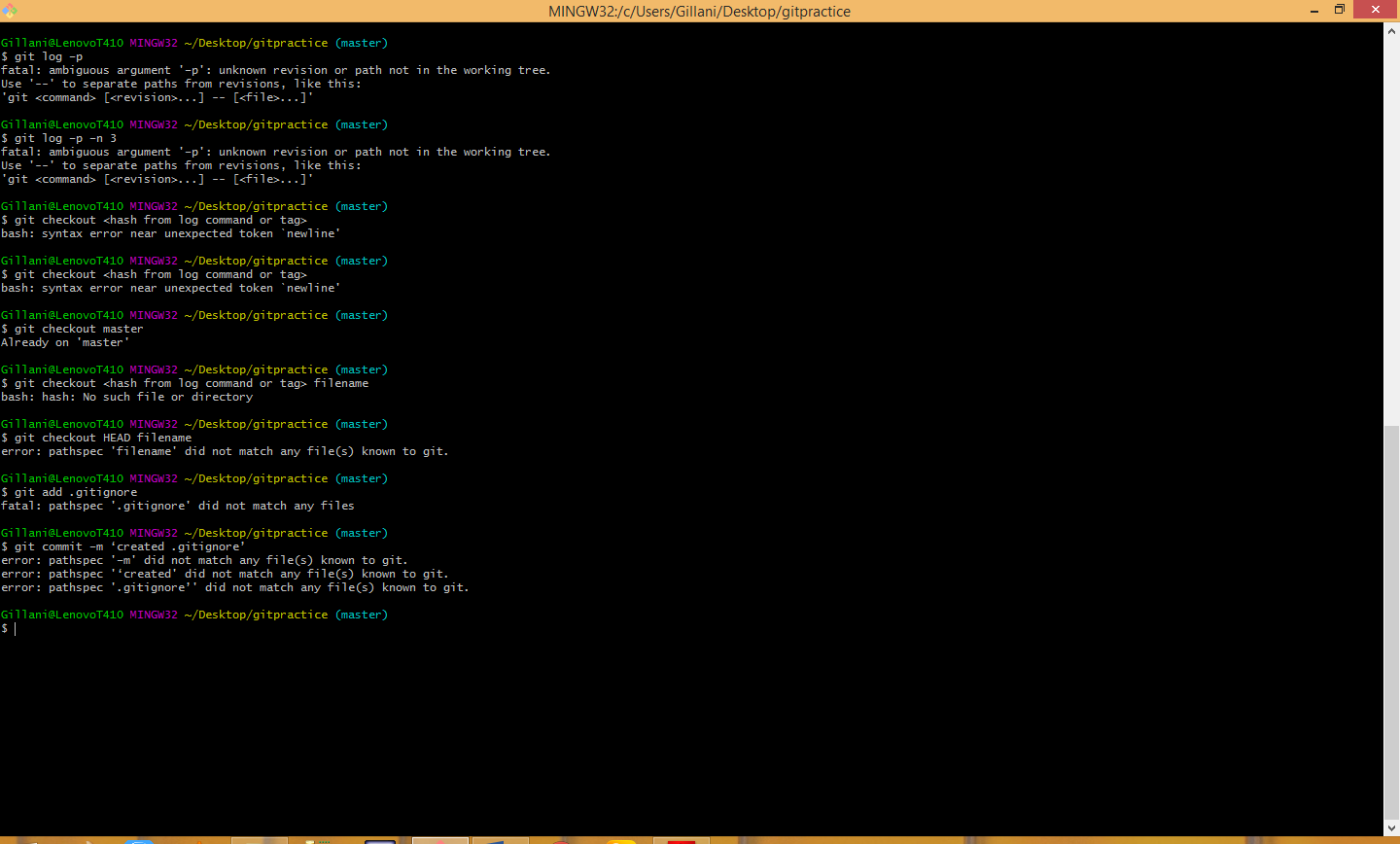
**$ git checkout HEAD filename**



Add and commit the .gitignore file to your project:

$ git add .gitignore

$ git commit –m ‘created .gitignore’

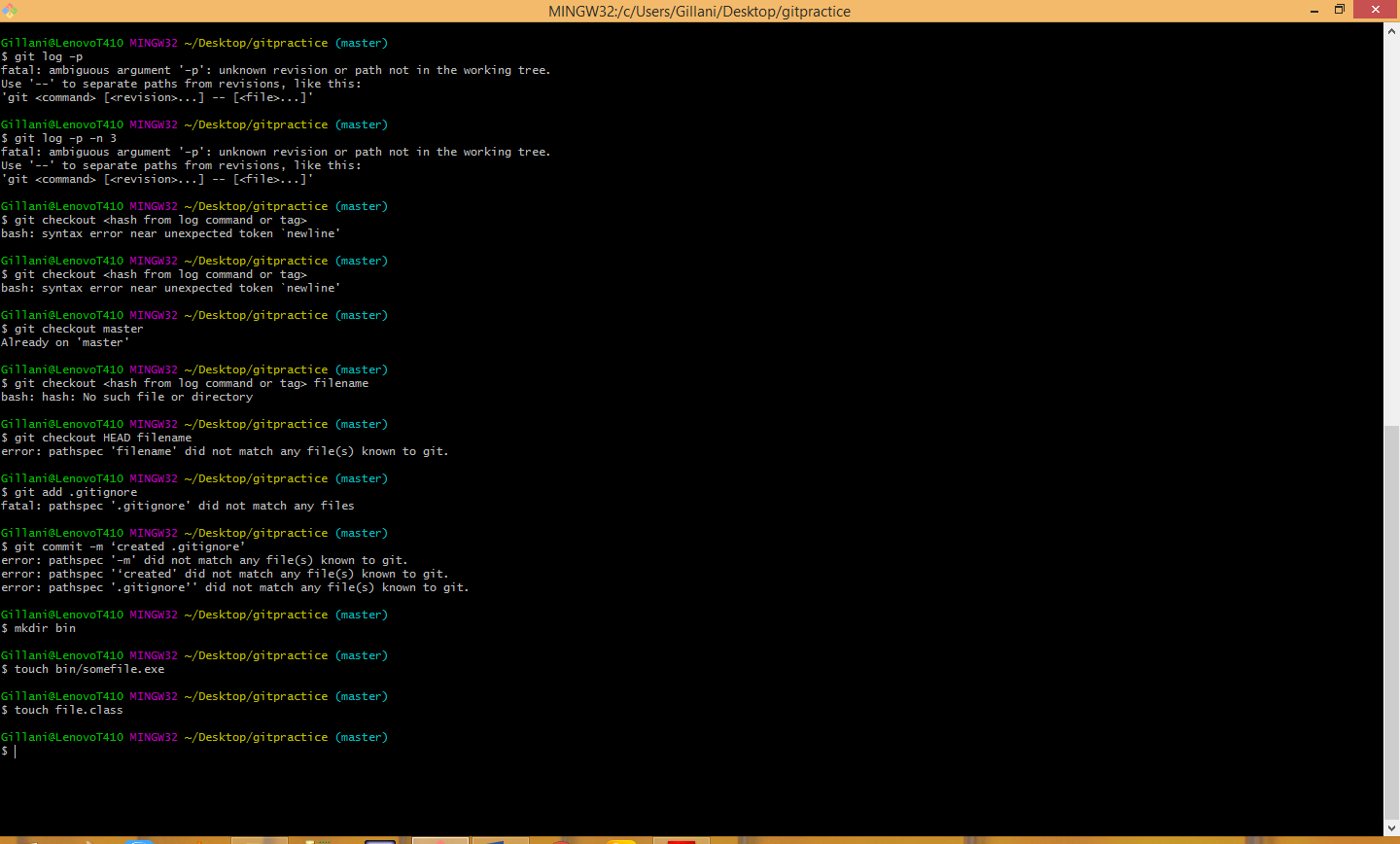


Now, create a bin directory, add a file to it and create a .class file:

$ mkdir bin

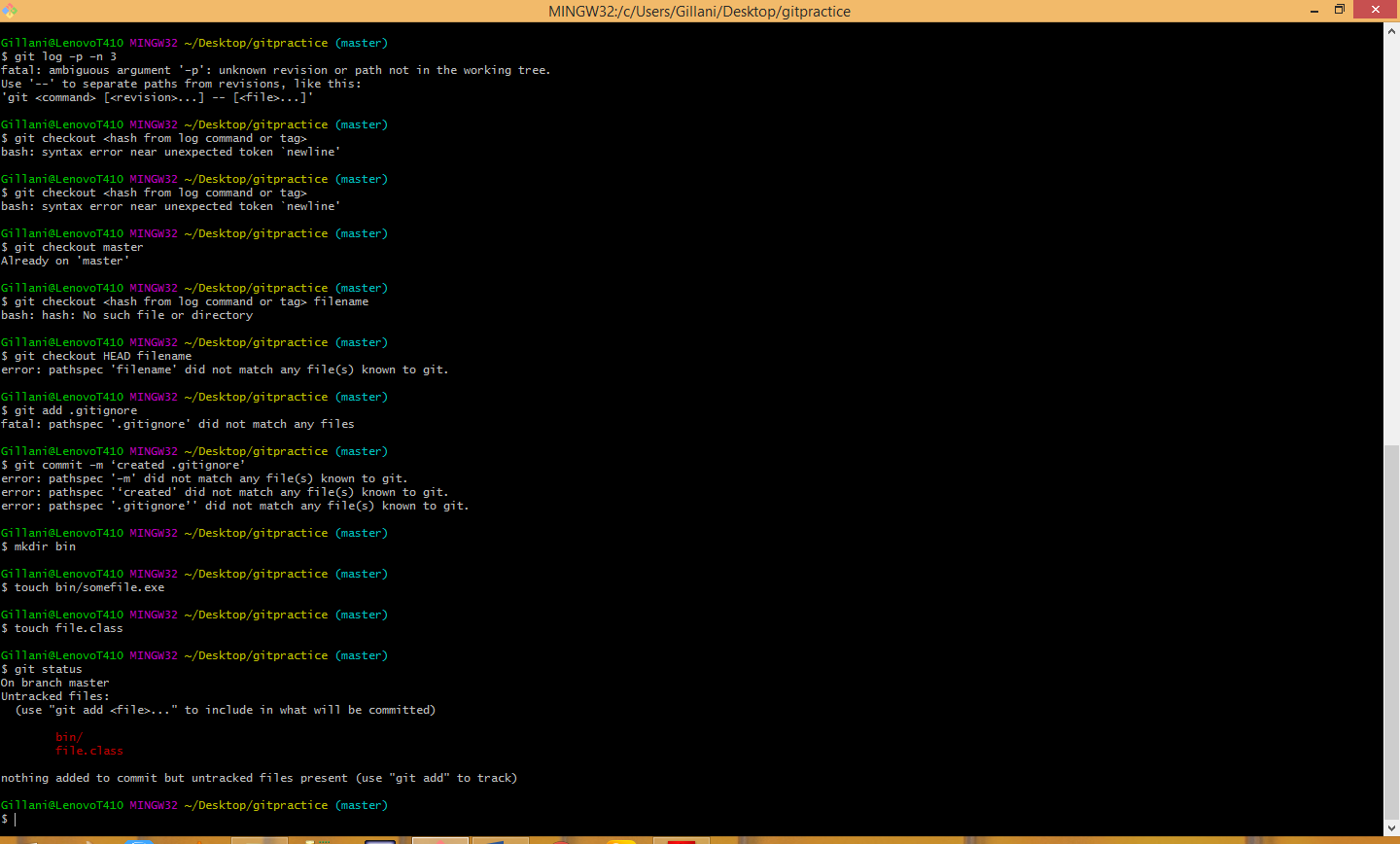
$ touch bin/somefile.exe

$ touch file.class



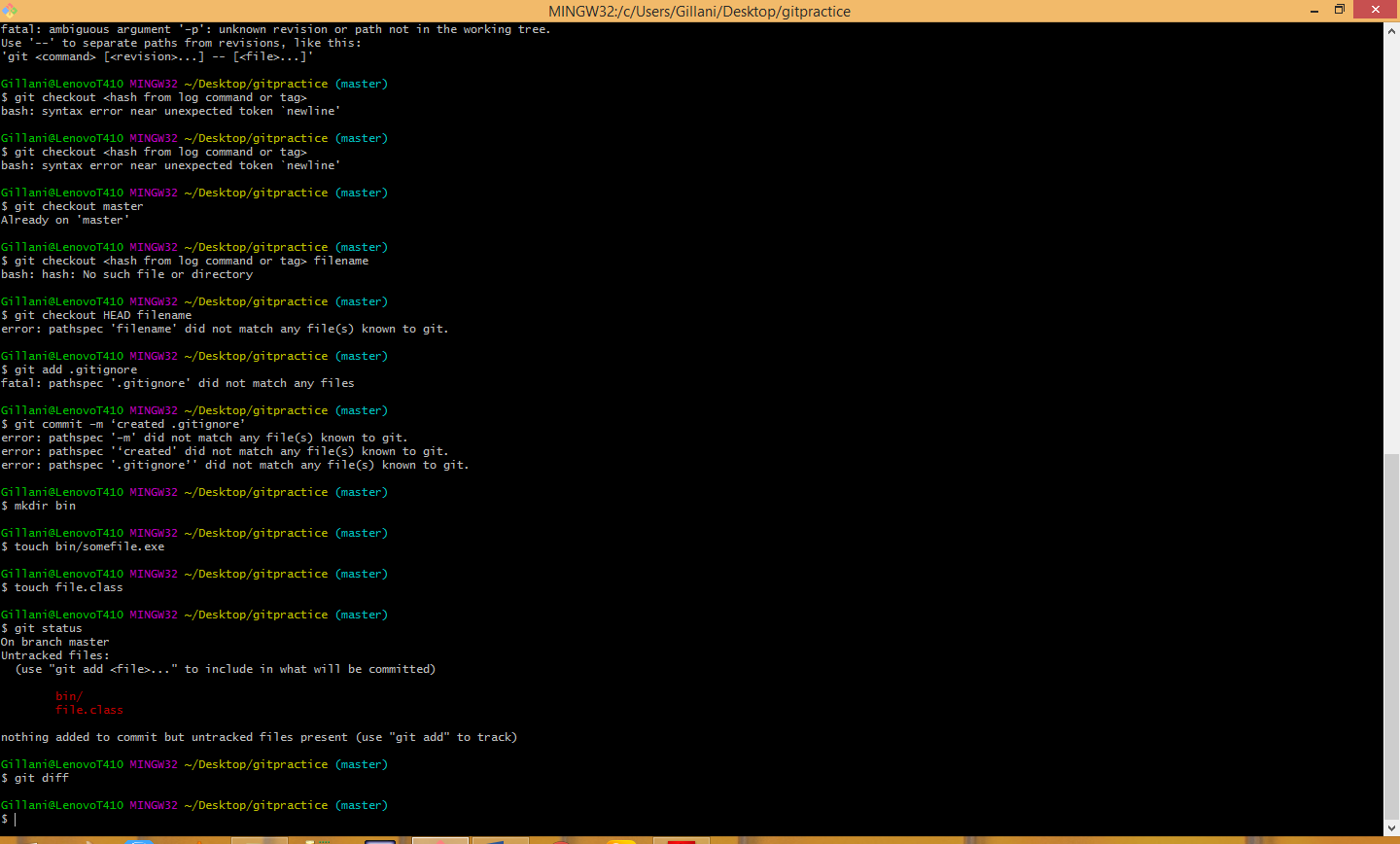
Now, check git status and notice these files are ignored (git doesn’t flag these new files as untracked):

$ git status



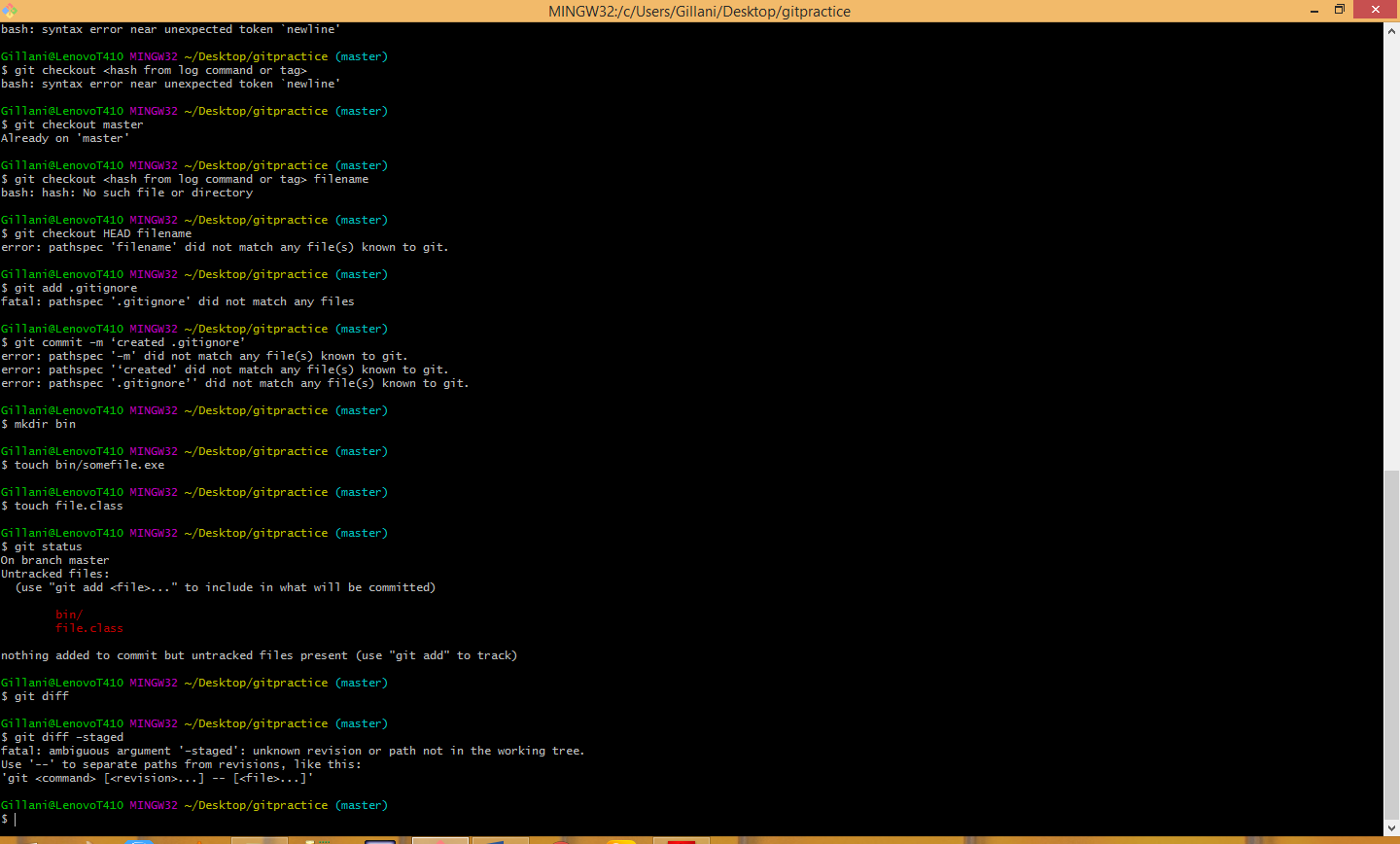
The following shows changed but not added:

**$ git diff**



The following shows exactly what will be added in the next commit (exact lines of each file):

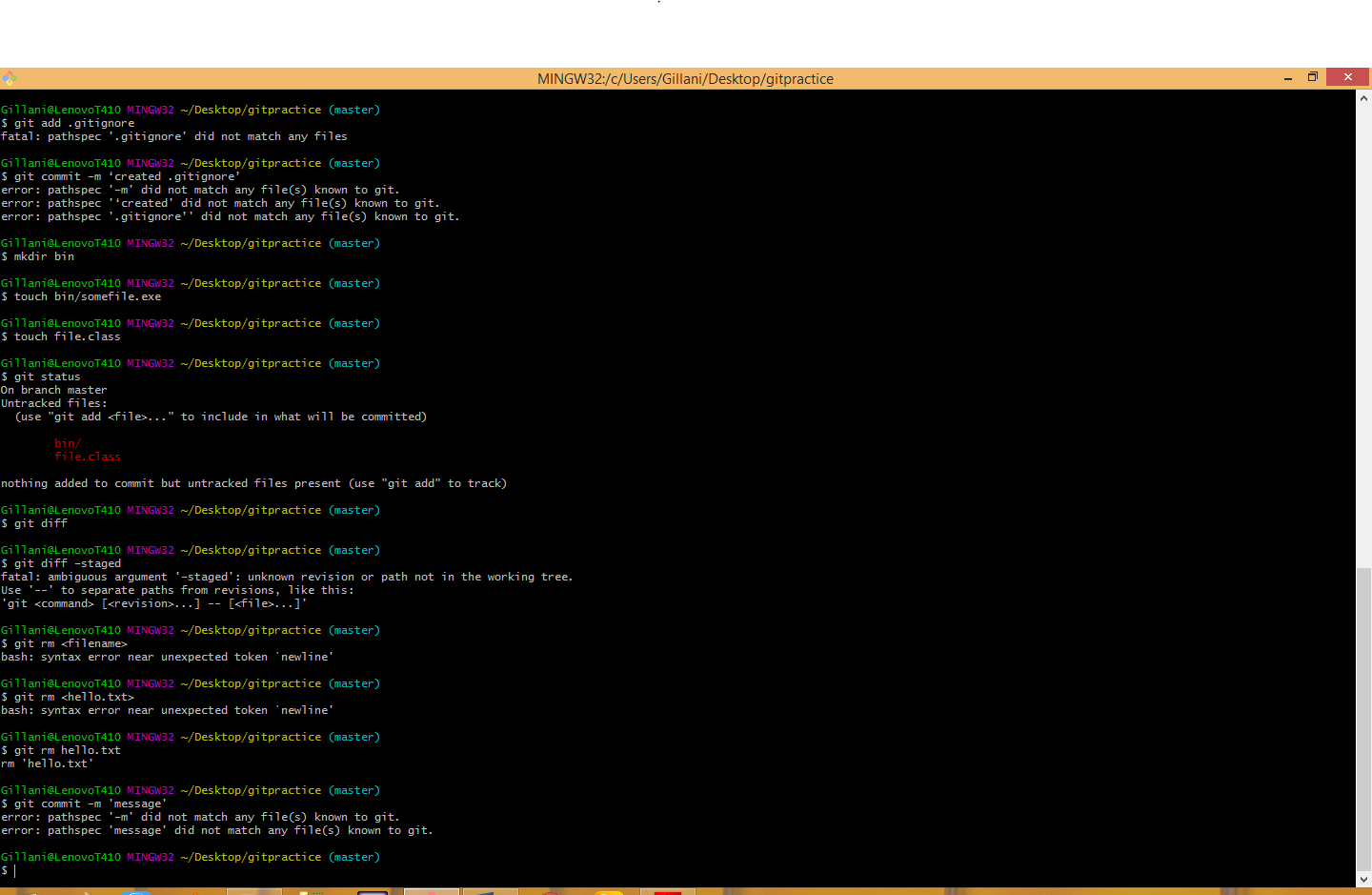
**$ git diff –staged**



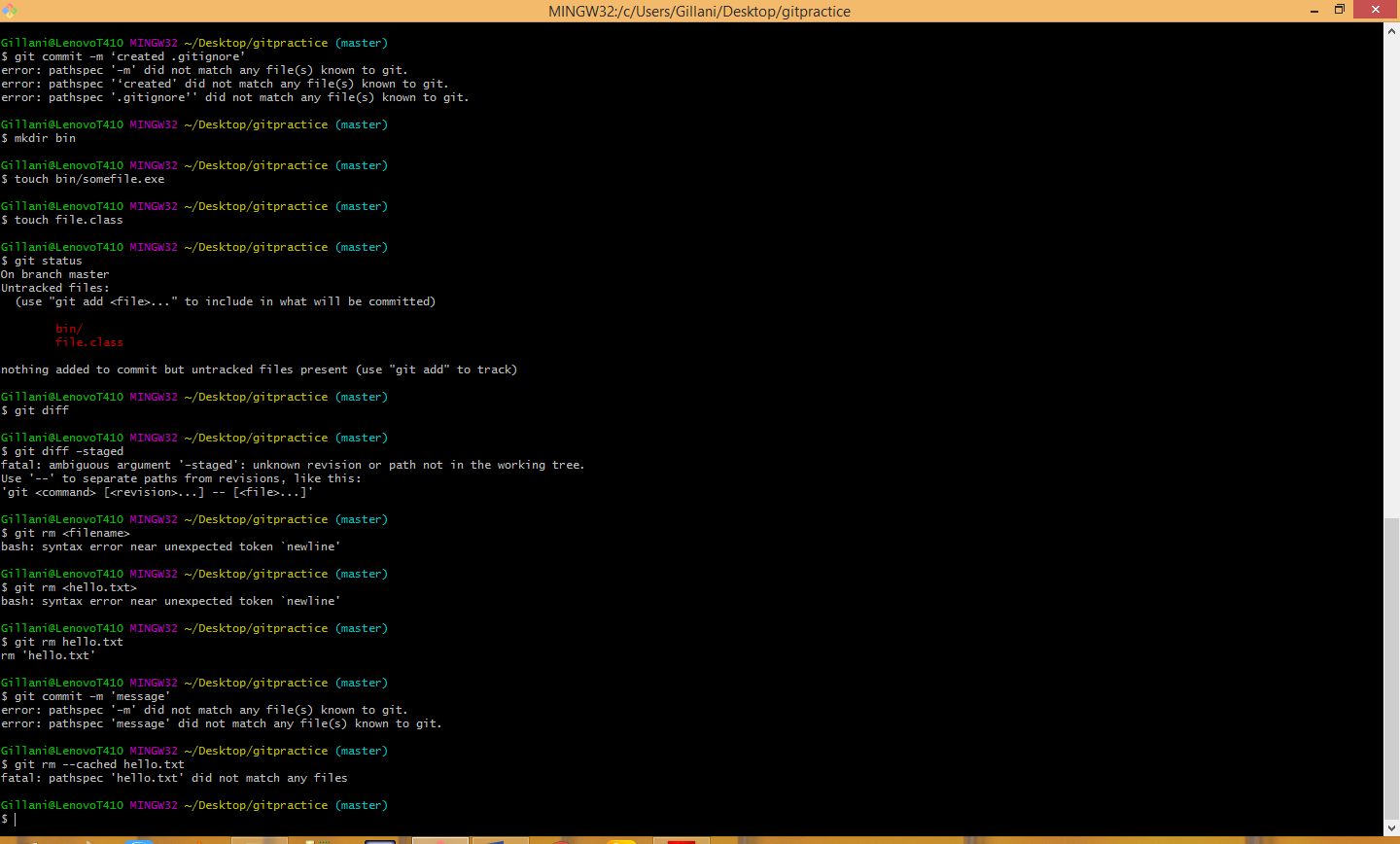
To delete a file from your project and remove it from being tracked, you can’t just delete and commit. To delete a file enter:

$ git rm <filename>

$ git commit –m ‘message’



**$ git rm --cached <filename>**



**$ mv oldfile newfile**



$ git rm oldfile

$ git add newfile

Or, you can accomplish the same using the git mv command:

$ git mv oldfile newfile