All commands seen here are run through GitBash

```
 git config --global user.name "[user_name]"
#tells Git who you are
```

```
git config --global user.email "[user.email]"
#tells Git your email address
```

```
git config --global core.editor "[your-editor-here]"
#If you want to change your default text editor for Git
#There are several different editors you can choose from such as
#notepad, vi, and visual studio code.
```

```
git init
#initializes a Git repository
```

```
git add *.py
#adds all files with [extension] to staging area
```

```
git add .
#adds all files within project to staging area
```

git status -s

#shows status of files to be committed

[-s displays a less verbose list of file statuses. If you want to see the full status and file name, leave the -s flag off]

git commit -m "init commit" [-m means "message".]

#Commits #changes and adds a message to each commit. For example, your #first commit's message might be "initial commit" or "init commit". Your #second commit message might be, "Added README and .gitignore #files".

git commit -a -m "[add-message-here]"

#stages and commits in a single command.

git [command] --help -or- git [command] -h

#shows you the help page for any command you want to use.

git rm --cached [filename]

#removes a file from git staging area and tells git not to 'track' it #anymore.

#Please note: If you do not include the --cached flag, you #will remove #the file from your working directory as well as from git.

git rm -r --cashed [directoryname]

```
#The -r flag means recursive. This command removes all the files
#from a particular directory from the staging area and tells git not to
#'track' them anymore.
```

```
touch .gitignore
#creates a special place for files you want git to ignore
```

```
nano .gitignore
#opens the 'nano' text editor. Once the nano text editor is open, add
#the file and/or directory names you no longer want git to 'track'.
```

```
git log –1
#shows the most recent commit message
```

```
git commit –amend
#opens the text editor (set by user during configuration) and allows
#user to change the most recent commit message. CTRL + S saves
#the file when done.
```

```
git commit --amend --no-edit
#commit the latest change without changing the commit message
```

```
git restore --staged <foldername>/filename
#Unstages files
```

```
git restore <foldername>/filename
#Discards changes made to a file
```

For GitHub Users:

Create GitHub account @ https://github.com

Create the first repository on the site. *It is up to you if you want to make it public or private. *

The following commands are used in GitBash Terminal:

```
echo "# [name-of-repository]">> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/<username-here>/<repository-name>_git.git
git push -u origin main
```

#If you wish to add a license.txt file and/or .gitignore file you will need #to input the following code into your GitBash terminal:

```
touch "license.txt"
nano "license.txt"
<add text here>
git add "license.txt"
git -s "license.txt"
git commit -m "[your-message-here]"
git branch -M main
git push -u origin main
```