Install GIT & make sure it is added into PATH.

Section 0 -Use GIT as local VCS. Steps to follow:

1.Create a directory ‘project_dir’ & cd to ‘project_dir’.

mkdir proj_dir & cd proj_dir

2.Initialize git version database. (git init)

$ git init
Initialized empty Git repository in C:/Users/DELL/proj_dir/.git/

3.Create a new file index.html.

4.Check the git status. You should find index.html as untracked file.

```
$ git status
On branch master

No commits yet

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

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

5.Stage the index.html file.

```
 git add index.txt
```

6.Commit index.html

```
$ git commit -m "commiting the index filee"
[master (root-commit) 071b819] commiting the index filee
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.txt
```

7.Make few changes in index.html & create a new file info.txt file.

8.Check git status. You should find index.html & info.txt as untracked files.
$git status
on branch master
No commits yet
Untracked files:

(use “git add <file>...” to include in what will be committed)
index.txt

(use "git add <file>..." to include in what will be committed)
info.txt

9.Configure GIT to ignore all txt files.

10.Again check the git status. You should find only index.html as untracked file.
Untracked files:

(use “git add <file>...” to include in what will be committed)
index.txt

11.State & commit index.html

$ git commit -m "commiting the index file"
[master a55c9d4] commiting the index file
1 file changed, 1 insertion(+)

12.Log all your comments so far.
$ git commit -m "commiting the index file"
[master a55c9d4] commiting the index file
1 file changed, 1 insertion(+)

DELL@shirisha-vanga MINGW64 ~/project_dir (master)
$ git log
commit a55c9d460f040a221873cd7e07b0b79ad79ee043 (HEAD -> master)
Author: ABCD <shirireddy10@gmail.com>

13.Make some changes in index.html.

14.Revert the change made in the previous step using git command.
$ git revert a55c9d460f040a221873cd7e07b0b79ad79ee043
[detached HEAD e761f3f] Revert "commiting the index file"
1 file changed, 1 deletion(-)

15.Again change index.html.

16.Stage index.html

$ git add index.txt

17.Revert back the last stage.

$ git revert fb87256bbdc1d7f2eee2ddbdb6be244330531d23
[detached HEAD be661c1] Revert "index file"
1 file changed, 1 deletion(-)

18 Rename ‘add’ command to ‘my-add’.
git mv add myadd

19.using my_add command stage index.html again &commit the changes.
git myadd
git commit-m “changes by add”
git commit

20.Revert the last commit
git reset HEAD

# *GIT Branching*

Objective: Commit HTML, CSS & JavaScript assignments into GIT.

SECTION-1 (HTML assignments) - Steps to follow:

First take a backup of your assignments & projects. This is required because due to incorrect GIT operation you may lose your files.
Create an empty directory ‘Assignments’ & cd to ‘Assignments’.
$ mkdir Assignments
$ cd Assignments

Create a file README.txt inside ‘Assignments’ & write few lines about the contents of ‘Assignments’ folder.

Commit README.txt file.
$ git commit -m "commiting the Readme file"
[master (root-commit) 21d9906] commiting the Readme file
1 file changed, 4 insertions(+)
create mode 100644 README.txt

Now create a new branch ‘html-assignments’.
$ git branch html-assignments

Switch to ‘html-assignments’ branch.

$ git checkout html-assignments
Switched to branch 'html-assignments'
git log --all --decorate --oneline --graph
* 21d9906 (HEAD -> html-assignments, master) commiting the Readme file

Copy all HTML assignments inside ‘Assignments’ folder.
Cp a1.html-assignments Cp a2.html-assignments

Commit HTML assignments into ‘html-assignments’ branch.
git commit -m “copy html files”

Make minor changes into few files belonging to ‘html-assignments’ branch.

Commit those changed files.
git commit -m “changes made in html-assignment branch”
Switch to master branch.

git checkout master
Make minor changes into README.txt file & commit those changes into master.

git commit -m “made some changes”

Again switch to ‘html-assignments’ branch.
git checkout html-assignments

Make minor changes into few files belonging to ‘html-assignments’ branch.
Commit those changes.
git commit -m “again made changes”
Switch to master.
git checkout master

Merge ‘html-assignments’ branch into master. Confirm all html assignments are shown in master.
git merge html-assignments

Finally delete the ‘html-assignments’ branch.
git branch -d html-assignments
Deleted branch html-assignments

SECTION-2 - (CSS assignments) Steps to follow:

1.Create a new branch ‘css-assignments’.
git branch css-assignments
2.Switch to ‘css-assignments’ branch.
git checkout css-assignments
3.Copy all CSS assignments inside ‘Assignments’ folder.
Cp a1.css-assignments Cp a2.css-assignments
4.Commit CSS assignments into ‘css-assignments’ branch.
git commit -m “adding css-assignments”
5.Make minor changes into README.txt file on line 1 belonging to ‘css-assignments’ branch.

6.Commit those changed files.
git commit -m “commiting with changed file”
7.Switch to master branch.
git checkout master
8.Make minor changes into README.txt file on line 3 & commit those changes into master.
git commit -m “commiting changes in master”

9.Again switch to ‘css-assignments’ branch.
git checkout css-assignments
10 Make minor changes into few files belonging to ‘css-assignments’ branch.
Commit those changes.
git commit -m “commiting with changes done to a css file”
11.Switch to master.
git checkout master
12.Merge ‘css-assignments’ branch into master. Confirm all css assignments are shown in master.
git merge css-assignments
13.Finally delete the ‘css-assignments’ branch.
git branch -d css-assignments

SECTION-3 - (JavaScript assignments) Steps to follow:

1.Create a new branch ‘js-assignments’.
git branch js-assigments

2.Switch to ‘js-assignments’ branch.
git checkout js-assignments
3.Copy all JavaScript assignments inside ‘Assignments’ folder.
cp a1.js-assignments cp a2.js-assignments
4.Commit JavaScript assignments into ‘js-assignments’ branch.
git commit -m “commiting with javascript changes”
5.Make minor changes into README.txt file on line 1 belonging to ‘js-assignments’ branch.

6.Commit those changed files.
git commit -m “commiting changes in readme”
7.Switch to master branch.
git checkout master
8.Make minor changes into README.txt file on line 1 & commit those changes into master.
git commit -m “commiting changes 2”
9.Again switch to ‘js-assignments’ branch.
git checkout js-assignments
10.Make minor changes into few files belonging to ‘js-assignments’ branch.
Commit those changes.
git commit “commiting changes into a js file”
11.Switch to master.
git checkout master

12.Merge ‘js-assignments’ branch into master. Confirm all JavaScript assignments are shown in master.
git merge js-assignments
13.Finally delete the ‘js-assignments’ branch.
git branch -d js-assignments

# *GIT Remoting*

Objective: Pushing source code into GITHUB & collaborate team members.

SECTION-3 (Pushing assignments to remote repository) - Steps to follow:

1.Create a github account if you do not have already.

2.Login on into github account.
logged into github account(shirireddy10)
3.Create new public repository ‘freshersbatch-oct16’.
https://github.com/shirireddy10/freshersbatch-oct16.git
4.Commit & push any sample file to this repository under ‘Assignments’ directory.
git commit -m “commiting the file”
git remote add origin https://github.com/shirireddy10/freshersbatch-oct16/index.git
git push origin master

SECTION-4 (Pushing source code to remote repository using Eclipse GIT plugin) - Steps to follow:

1. One developer from project team will create eclipse projects ‘SampleProj’ & add sample source code files. Then commit all files through eclipse GIT plugin.
   a.open eclipse IDE then shift to git repository
   b.select add on existing local repository
   c.Browse the repository folder and add to it
   d.select to clone a repository and enter our github url and finish
2. Collaborate other team members with your github account so that they can also modify the committed files.

3. Other developers from same team will checkout all files from remote repository. This might get conflicts since certain files fail to merge. In such case, merge it manually.

4. Commit & push the ‘SampleProj’ project.
   a.Right click on project and select commit

b.select commit

c.we can git staging view then select commit

d.then right click on the project and select team

e.In that team menu select push branch master