DSI: Unix Shell, Git and GitHub
Assignment 2 & Quiz: Git and GitHub –
Apichat Photi-a

1. Check all that are TRUE about version control:
   - [X] True > Can revert files to a previous state
   - [X] True > Can compare changes over time
   - [X] True > Can see who modified something last
   - [X] True > Can recover lost files

2. What is the difference between centralized version control systems and distributed version control systems?
   Ans In a centralized version control system (CVCS), there is a single central server that contains all file versions from developers being managed. Developers check out a copy of the file from the central repository, make their changes, and then check the file back into the central repository. The CVCS manages conflicts and merges between different versions of the file.

   A distributed version control system (DVCS) allows developers to create their own local repositories containing a complete copy of the project's version history. Dev can make changes to their local copy and commit them to their local repository. They can also pull changes from other devs' local repositories and send their own changes to other dev's local repositories. This means each developer has their own complete copy of the project history, and the system manages to merge changes between these copies.

3. What are the three states that files can reside in?
   - ☐ committed, changed, waiting
   - ☐ saved, changed, staged
   - X ☐ **committed, modified, staged**
   - ☐ saved, modified, staged

4. What command initializes a new repository?
   - ☐ `git clone`
   - ☐ `git branch`
   - ☐ `git fork`
   - X ☐ **`git init`**

5. What does `git diff` do?
   - ☐ compares the differences between the home directory and staging arearuler
   - X ☐ **compares the differences between the working directory and staging area**
   - ☐ compares the differences between the working directory and what's been committed
   - ☐ compares the differences between the staging area and what's been committed

6. How do you add a message to your commit? (select all that apply)
   - X ☐ **`git commit -m`**
   - ☐ `git commit -messages`
   - ☐ `git commit`
   - ☐ `git commit -message`

7. How do you add a remote repo? (select all that apply)

X ☐ **`git remote`**

☐ `git add remote`

X ☐ **`git clone`**

☐ `git add clone`

8. What is the difference between `git pull` and `git fetch`?

Ans The main difference between "git pull" and "git fetch" is that "git pull" automatically merges changes from the remote repository with local branch, while "git fetch" only downloads the changes, leaving it up and we need to merge them manually with "git merge".

9. How do you switch branches?

X ☐ **`git checkout`**

X ☐ **`git checkout -b`**

☐ `git branch -c`

☐ `git branch`

10. Why are messages important? What would make a good commit message?

Ans Commit messages ate important because they provide context and documentation for the changes made in a commit. A good commit message should be informative, descriptive, and easy to understand. It should provide enough context about the changes made in the commit, that other contributors can understand the changes easily but not too long.

11. Please correct the merge shown below (both codes are suitable, neither has errors):

```
<<<<<<< HEAD
df.loc[df['sex'] == 'f', 'age'].mean()
git commit -a
=======
df.loc[df['sex'] == 'm', 'age'].mean()
>>>>>>> branch_1
```

Ans

Calculate the mean age for both male and female sex groups.

```
mean_age_f = df.loc[df['sex'] == 'f', 'age'].mean()
mean_age_m = df.loc[df['sex'] == 'm', 'age'].mean()
```

then commit the changes

```
git commit -a -m "calculate the mean age for both sex gr"
```

Part 2

- [x] 1. `fork` and `clone` this class GitHub repo.
- [x] 2. `push` your Assignment 1 to the folder labelled “assignment-2.” Your additions should include…
  - All components necessary to run Assignment 1
  - Proper folder structure (inputs, outputs, scripts)
  - A README.md file. The README should include components discussed in the workshop. Feel free to research good READMEs and add anything that you believe will add value to your README
- [x] 3. Create a `pull request` to add your additions to the class repo.

Rubric:

Component	**1**	**2**	**3**	**4**	**5**
1. Repo contains all necessary components to run Shell script					
2. README is comprehensive and includes components discussed in class plus at least one component learned from outside sources					
3. Pull request has been successfully closed and pull request to partner contains at least one helpful addition					
Total:					**/15**
Quiz Total:					**/11**
Final:					**/26**