1. **GIT INSTALLATION PROCEDURE:**

**For Linux based system:**

Open the Terminal and type “sudo apt-get install git”

**For Windows system:**

Download git bash and install it.

GUI is also available for windows based systems.

1. **BASIC GIT COMMANDS:**
2. **git init:** Initializes an empty git repository. This is a one-time command that creates ‘.git’ sub directory under the present working directory. This also creates master branch
3. **git clone:** It copies the existing project into local repository from a remote repository.
4. **git add:** It performs staging. Staging is a process where the changes are about to be committed to the project. This gives us a control over the changes that we are sure about commitment to the project.
5. **git commit:** Makes the changes to the project and creates a version of the project. The project history can be tracked using the history of commits. Anything that is staged with git add command will be the part of this commit.
6. **git status:** Shows the details of the untracked files, staged files and already committed files in the working directory.
7. **git branch:** Shows the branch that we are working on, locally.
8. **git merge:** When we work on a project in git environment, instead of working directly on the master branch, we can create the copy of the master branch and perform our experiment on the copied branch. Once the experiment is successful, we can merge the branch into the master branch. This is used especially while developing extra features to the existing project.
9. **git pull:** When others have made changes to the remote repository, they do not reflect on our local repository. To make them get reflected on our local repository, we need to pull the remote repository.
10. **git push:** When we make changes to our local repository, they do not get reflected on the remote repository. We should push the committed changes to get reflected on the local repository.
11. **USING GIT - PROCEDURE:**
12. Create directory on the local system and initialize an empty repository using ‘git init’.
13. Configure the username and email-id of github to make further things in the repository. Use ‘git config user.name=”username”’ and ‘git config user.email=”mailid”’
14. If the project exists in the remote repository, copy the repository into the local system using ‘git clone “url of remote repository”’
15. Make the changes to the repository as per requirement and stage the required changes to the commit using the command “git add filename”. If all the files that we added to the repository are to be staged, use “git add .”.
16. Commit the changes to the local repository using “git commit –m “message required””
17. Add the origin of the remote repository using “git remote add origin “url of the remote repository””.
18. Push the changes that are made in the local repository onto the remote repository using “git push origin master”
19. **GIT BRANCHING - PROCEDURE:**

Branching is a process of pointing to the commit. It doesn’t create a copy of the project but gives us the privilege to experiment adding new features to the existing project without disturbing the core project. Once the experiment is successful, we can add the feature to the master branch by merging.

1. **git branch:** “git branch “BranchName”” creates a new branch.
2. **git checkout:** The previous command just creates a new branch. But it doesn’t handover the control on to the new branch. So whatever the changes we are doing, they are being done on the master branch itself. So to move the control to the new branch, we use “git checkout “BranchName””. Now the commits we make, will be made on this new branch.
3. **git merge:** After successful completion of commits on the new branch, we need to merge the branch into the master branch. So shift the control on to master branch once the work is completed in the working branch using “git checkout master”. Now merge the worked branch into the master branch using “git merge “Branch that has to be merged””.

Now the resulting commit has two parents, one is the master and other is the branch that we worked on. And now master has all the files in the repository. However, the branch doesn’t have the commits made by master. To do this, shift the control on to the branch and merge master using “git checkout “Branch Name”; git merge master;”

1. **git rebase:** Another way of merging the branches is rebasing. Move on to the new branch and rebase the working branch on to the master. “git rebase master”