# GITHUB

1)Download and Install the Git from this url **https://git-scm.com/download/win**

2)Terminal is another way to interact with the computer. It give much understanding of the git processes.

**Below are basic Terminal commands:**

**PWD** – Present Working Directory will be printed

**LS** – It will list all the files present in the directory

**CD** – Change directory

**LS -L** – It will list all the file with the detailed information like time etc

**CD ..** – It will back to previous folder which might be its parent

**CD ../..** – It will back two time to previous folder which might be its parent

**CD~** -- It will get back to home

**CD /** -- It will navigate to the root directly of the computer

**Mkdir foldername** – creates a new folder

**Touch filename.format** – Creates a new file

**Rmdir foldername** -- Deletes the empty folder

**Rm -r filename** – Deletes the folder with contents as well

**Rm filename.format** – Deletes the file

**Echo "aarthi"** – It will print the contents on the console

**Cat > filename.format** – we can create a file using this and write the contents. Ctrl+D will do save the content.

**Cp filename1.format filename2.format** – It will copy the contents of the file1 to file 2

**Cp filename.format filepath** – It will copy the file and paste them in to the given path

**mv filename.format filepath** – It will move the file and paste them in to the given path

**ls | grep "prakash.txt"** – It is a search command

**date** – Displays date, time, time zone, etc.

**ps** – Lists the running processes in the system

**kill <process id>** – Used to kill a process

### 3)Git Commands

**Git init** → It will initialise a empty repository in the working folder. A .git hidden folder will be created in the working folder that keeps track of all the files, folders and changes in your project

**Below commands are used to set the credentials to the repositories:**

git config --global user.email "prakashm75101@gmail.com"
git config --global user.name "prakashm75101"
git config --global user.password "Prk7596@"

**Git Status** → It will display the status of the working branch. By default it will be Master Branch and then it will show the Untracked files(that is not stages or newly added) and also not committed files.

**Git add** → It will add the files to the staging area from the current directory.

**Git log** → It will show the commit logs.

**Git commit -m "commit message"** → It will commit the code to the staging are. Staging is the temporary storage in the pc and then commit code will get generated like this c4b95444486fea4674dbd88493605a44a192cb3e using this we can revert in future.

**Git revert -n commitid fully or 7 characters** → It is used to revert the changes done by the given commit. **-n** is used to hold the commit so that after reverting we need to again give **Git commit -m "commit message"**

**Git reset commitid - -hard** → It will reset the commit as well as the file to the given commitid version. Commit logs will be removed.

**Git reset commitid** → It will reset the commit alone to the given commitid version. Commit logs will be removed.

**Git revert head** → It will rever the most recent commit.

**Git reset head~1** → It will reset to 1 commit before the most recent commit.

**Q** → It will allow to enter when the git is showing press return button.

**Git branch branchname** → It is  used to create a new branch. Before creating any new branch we need to do atleast one commit. Otherwise it will show the fatal error in the console.

**Git branch** → It will list all the available branches

**Git checkout branchname** → It is used to switch to the given branch name.

**Git checkout -b branchname** → It is used to create and switch to the given branch name.

**git push --set-upstream origin branchone** → It is used to push the code to the repository if the master is not there

**Git Push** → It is used to push the code to the current branch

**git remote add origin https://github.com/Prakashm75101/testrepo.git** --> It is used to add the remote path

**Git remote** → It will display the available remote branches

**Git push origin master** → It will push the code in the master i.e., local to the remote origin branch.

**Git push -u origin master** → It will track and push the code in the master i.e., local to the remote origin branch. After giving this command we can just use Git Push command alone.

**Git push –all origin** → It will push all the branches to the repo

**Git pull** → It will pull the repo code to local

**Git fork** →Is make another remote repository due to access priviledges.

**Git Merge branchname** → It will merge the current branch to the given branch name

**GIT diff filename** → It will list the changes done to the particular file. Green line with plus symbol indicates the line has been added. And red line with minus symbol indicates the line has been deleted.

**git clone --branch branchone https://github.com/Prakashm75101/testrepo.git-->** It is used to clone the remote branch to the local machine.

**Fast forward** → indicates we don't have any conflicts

**Git stash save** → It is used to save the local changes temporarily while pulling the code from the repo if there is any conflicts.

**Git stash pop** → It is used add the code back to file along with the pulled data.

Git rebase branchname → It will change the base of the branch from one commit to another.

**Git squash** → It is used to rewrite the commit history. So that we can make some clean up on commits.

**GIT Questions:**

**1)Difference between git and github?**

Git is a version control tool whereas github is the repository

**2)Difference between revert and reset?**

Revert is used to revert the changes done by the given commit and it wont remove the commit logs.

Reset is used to revert the changes done by the given commit and it will removes the commit log as well. Reset will be typically used to remove the sensitive information like passwords from the commit logs.

**3)How to resolve conflicts?**

In case of conflicts while pushing the code, I will first pull the code from the master and then resolve the conflicts. Later I will add, commit and push the code again.

**4)How to ignore a file to staging?**

Create a .gitignore file in the project. In this one we can add that file name with format. After this if we give git add . Then added file name in .gitignore file will get ignored.

**5)Difference between git rebase and git merge?**

git rebase creates a new set of commits applied on top of the target branch, while git merge creates a new merge commit that combines the changes from both branches.