**First time adding files to repository**

**echo "# test" >> README.md**

**git init**

**git add README.md**

**git commit -m "first commit"**

**git commit –m “ticket : 101” -- in real commit message will be like this**

**git remote add origin https://github.com/repo/repo.git**

**git push -u origin master**

**To remove origin**

**git remote rm origin**

**First create/changes the files**

**git add \***

**git add --all or -A** --If we have multiple dir. It will add all dir

**git commit -m message**

**git commit -a -m message** ---- it will add the files to staging area and commit the file

**git commit --amend -m “message”**  -- to change the commit message of already committed message

**git push -u origin master**

**Clone**

**For Http link**

**git clone** [**https://github.com/repo/repo.git**](https://github.com/repo/repo.git) **test --** project repo will be copied into test dir in local machine

**For SSH link**

**git clone username@servername(192.168.10.1):repo name(project.git)**

**git checkout test** -- it changes branch

**git status**

**git branch test** --- it will create test branch

**git branch** --- it will show the current branch and list all branches

**git checkout -b test\_branch** --- it will create new test branch and change the present branch

**git checkout test** --- it will change the branch

**git branch -D test\_branch --** it will delete branch

**git branch -m <old branch name> <new branch name> ---** It will rename the branch

**git branch -r** **---** It list out all remote tracking branches

**git branch -a ---**  It list out all branches

**Then change the file content**

**git add \***

**git commit -m "clone"**

**git push -u origin test**

**For merge**

**git checkout master ----** it changes the branch from old to new

**git merge test --** You have to be in master branch to execute this command

**git status**

**git pull origin**

**git push -u origin maste**r

**git diff master -- test** -- Shows content differences between two branches.

**git diff pwd** --- It shows the content diff in files

**git config --global user.name “user” user.email user**[**@gmail.com**](mailto:vamshi@gmail.com) --- to set name and mail id

**git config --list** --- it will list all default parameters

**cat ~/.gitconfig** -- It shows the configures username and mail

**git config –system** **user.name “user” user.email user@gmail.com** -- to set name and mail id at system level.

**git config –local user** **user.name “user” user.email user@gmail.com** – to set name and mail id at project level.

**ssh-keygen -t rsa -C “ssh keys”** -- to generate keys

**git -rm --cached <filename>** ---- it will untrack the file or it will remove file from staging area

**git remote -v ---** which list all remote repositories **OR** Verify the remote

→ If you write an file name in .gitignore file. That file will never committed,even if you are trying commit forcefully.

**git log --**  it will show log all commits

**git log -2** -- it will last to log entries

**git stash --** push a new stash onto your stack

**git stash list --** We can view a list of stashed changes by using the git stash list command.

**git stash pop --** to remove the changes from the stack and place them in the current working directory.

**git mv string.c src/ ---** To move the data from present working dir to src dir

**git mv string.c string\_operations.c ---**  To rename file

**git rm string\_operations ---** remove the file from repository

**git checkout <File name> -- U**ndo the changes made to file

**git log** --- It shows the commit ids.

**git diff <commit id> ---** It shows the modified content in file

**git checkout <commit id> --filename --** to undo the changes to that commit id

**git status -s   
Output: D string\_operations.c --**  git shows D when u sue check out

we can use the **git checkout** command to obtain a deleted file from the local repository.

**git checkout <File name>**

**Note:** We can perform all these operations before commit operation.

→ In Git, there is one HEAD pointer that always points to the latest commit. If you want to undo a change from the staged area, then you can use the git checkout command. but with the checkout command, you have to provide an additional parameter, i.e., the HEAD pointer.

**git checkout HEAD -- string\_operations.c**

**git reset --soft HEAD~ ---** Each branch has a HEAD pointer, which points to the latest commit. If we use Git reset command with --soft option followed by commit ID, then it will reset the HEAD pointer only without destroying anything.

→ Git reset with **--mixed** option reverts those changes from the staging area that have not been committed yet.

→ If you use **--hard** option with the Git reset command, it will clear the staging area. it will reset the HEAD pointer to the latest commit of the specific commit ID and delete the local file changes too.

**git reset --hard 577647211ed44fe2ae479427a0668a4f12ed71a1**

# Git - Tag

→ Tag operation allows giving meaningful names to a specific version in the repository.

**git tag -a 'Release\_1\_0' -m 'Tagged basic string operation code' HEAD**

**git push origin tag Release\_1\_0**

→ If you want to tag a particular commit, then use the appropriate COMMIT ID instead of the HEAD pointer.   
**git tag -l --** It shows all tag information

**git show Release\_1\_0 --**  It gives particular tag information

**git tag -d Release\_1\_0 --**  To delete tag

**Git Repo Backup**

**→ git bundle create /path/to/mybundle master branch2 branch3 ----**  Git bundle let you pack the references of your repository as a single file, but unlike the tar command above the bundle is a recognized git source. You can fetch, pull clone from it.

→ **git bundle list-heads /path/to/mybundle ----** You can then check what is in your bundle

### Generating a new SSH key

**$ ssh-keygen -t rsa -b 4096 -C "**[***your\_email@example.com***](mailto:your_email@example.com)**"**

.git/config

[remote "origin"]

fetch = +refs/heads/\*:refs/remotes/origin/\*

url = git@bitbucket.org:emmap1/bitbucketspacestation.git

**$ git remote add origin remote repository URL**

# Sets the new remote

**$ git remote -v**

# Verifies the new remote URL

git push origin master  
# Pushes the changes in your local repository up to the remote repository you specified as the origin

**Commands in hand**

→ git --version

→ git config --list

→ git config --global user.name “user"

→ git config --global user.email "user@gmail.com"

→ mkdir c:\temp\gitclass

→ mkdir c:\gitkrishna\gitclass

→ cd c:\gitkrishna\gitclass

→ git init

→ vi file1.txt

→ git add \*

→ git status

→ git commit -m "initial commit"

→ git remote -v

→ git remote add myorigin https://github.com/user

→ git push -u myorigin master https://github.com/user

→ git status

→ git commit -m "second commit"

→ git commit -m "third commit"

→ git status

→ git checkout vamshi

→ git remote add origin https://github.com/krishnakvk/project.git

→ git status

→ git push -u origin vamshi

→ git push -u origin vamshi https://github.com/krishnakvk/project.git

→ git remote rm origin

→ git remote add origin https://github.com/krishnakvk/project.git

→ git push -u origin vamshi https://github.com/krishnakvk/project.git

→ vi file1.txt

→ git status

→ git commit -m "fourth"

→ git add \*

→ git commit -m "fourth"

→ git status

→ git push \*

→ cd c:gitkrishna

→ git add \*

→ git commit -m "Initial commit"

→ git remote add origin https://github.com/krishnakvk/krishn.git

→ git push -u origin master

→ git checkout master

→ git \*

→ ls -l .git/refs/heads/

→ cat .git/refs/heads/master

→ git -v

→ cd refs

→ git config --list

→ git diff file1.txt

→ vi file1.txt

→ git diff file1.txt

→ git add \*

→ git commit -m "fifth commit"

→ git push -u origin master

→ git clone \* c:\test

→ git clone /c/Users/DAY/c:gitkrishna

→ git class /c/Users/DAY/c:test

→ git clone test

→ git clone /c/Users/DAY/c:gitkrishnagitclass c:/test

→ git checkout temp

→ git clone https://github.com/krishnakvk/project.git test

→ git checkout test

→ git status

→ git branch test

→ git status

→ git checkout test

→ vi file1.txt

→ git add \*

→ git commit -m "clone"

→ git push -u origin test

→ git checkout master

→ git merge test

→ git status

→ git push -u origin master