* ‘Git status’ – To check the status of files. It will tell about files in the repository but are not tracked
* ‘Git diff filename / directory / without filename’ – To check difference between current state and what was saved last time.
* ‘git diff -r HEAD filename’ -r means ‘compare to a particular version’ and HEAD means ‘the most recent commit’
* Use tools like DiffMerge or WinMerge to see the changes side by side
* To add files to the staging area, use ‘git add filename’. Files in the repository are by default not tracked. Use git add to start tracking the file. There can be files changes which are staged and changes which are not staged. To save the changes to staging – git add filename
* File editor (Nano)
  + Nano filename – it will open the file
  + Ctrl-k to delete a line
  + Ctrl-u un-delete a line
  + Ctrl-o save the file
  + Ctrl-x exit the editor
* To save changes in the staging area, git commit. Saves everything in staging area as one unit
* Git commit -m “message”
* Git commit –amend -m “new message”
* Git log – To view the project’s history. (Most recent first). Use space bar to move to next page and q for quitting
* Commit consists of metadata, Tree and blob
* Every commit will have a unique identifier called Hash id.
* To view details of a specific commit use ‘git show first\_few\_characters\_of\_hashId’
* Git show head – Will show the recent commit, git show head¬1 will show one commit before recent. (There should be no gap between tilde and after)
* Git annotate file – This gives more info (who made the last change to each line of a file and when)
* To see difference between commits – git diff ID1..ID2 (or) git diff HEAD~1..HEAD~4 (Two dots .. are connectors)
* To ignore tracking files – create .gitignore in root directory and storing a list of filenames using wildcard to ignore the files
* Git clean -n (list of files whose history git is not currently tracking)
* Git clean -f (delete the files)
* To see the git settings
  + Git config –list –system (settings for every user on this computer)
  + Git config –list –global (settings for every one of your project)
  + Git config –list –local (settings for one specific project) (Highest privileages)
  + Git config –global setting value (To change name and email address)
  + User.name & user.email are the settings
* To stage a single file – git add path\_to\_file
* To unstage the additions git reset HEAD
* To undo changes to unstaged files
  + Git checkout – filename (recovery is not possible)
* To undo changes to staged files
  + Git reset HEAD path\_to\_file
  + Git checkout path\_to\_file
* To resore a old version of the file
  + Git checkout Hashid filename

Branching:

* Git branch – this will show all the branches, the branch you are working will be shown with a \*
* Git diff branch-1..branch-2 will show the difference between two branches
* Git checkout branchname – To switch to the branch
* Git rm – Removes the files and stages the removal of that file in single step
* Git checkout -b branch-name - To create a branch and switch to a branch
* Git merge source destination

Creating a new repository in the working directory:-

Git init project-name

Turn existing project into a git repository – git init

The original site from which a repo is cloned is called remote

Git remote -v – This will provide the information about remote URL (origin is the original repository)

Add remote using – git remote add remote-name URL

Remove remote using – git remote rm remote-name

Updating local with remote – git pull remote branch (This updates the current branch in local repo)

To make changes to the remote after working on local – git push remote-name branch-name