GIT commands that are important to learn to use command line

1. **git config –global user.name Anurag** – For configuring the name
2. **git config –global user.email beaprogrammer24@gmail.com** – to set the email
3. **git config –global user.name** – For checking the changes have been affected
4. **git config –global user.email** - For checking the changes have been affected
5. **code .** – to open the VS Code
6. **git init** – initialize the git repository
7. **ls -lart** – To see the Hidden files
8. **git status** – To check the status of the files
9. **git add index.html** – To add to the staging area
10. **git commit** – to send the staged files to commit
11. **press Insert then :wq** then write the name of the initial commit (Wim editor)
12. **touch about.html** – to create a empty file named with about.html
13. **git add -A** – To add all the untracked files to staging area
14. **git commit -m "Any Commit message"** – to add the commit and message **shortcut**
15. **clear** – to clear the previous files
16. **git checkout contact.html** – To match the file with last commit
17. **git checkout -f** – All files get matched to the previous files
18. **git log** – to check the activity of the files, all the commits are shown and config files
19. **git log -p -1**– to see he last 1 commits on the machine and shows the changes we made
20. **press q to quit**
21. **git diff** – it compares the working tree with staging area(if both are same then no output)
22. **git diff –stagged** -compare the last commit with staging area
23. **git checkout -f** – matches the file with last commit
24. **git commit -a -m "skipped staging area and fixed"** -If you wat to directly commit the file without making it to the staging area(for beginners only)
25. **ls** -we can see all the files as it is feature of UNIX.
26. **touch delete.html -**to create a empty file
27. **git add -A**
28. **git commit -a -m "Add waste file"**
29. **git rm –cached** – it will remove from staging area only not from hard disk
30. **git rm waste.html** -If you want to delete a file
31. **git commit -a -m "removing waste"**
32. **git log -p -2** – If you want to see 2 commits
33. **git status -s** – shows the summarized status(modify the file then use this command)
34. git add file.html
35. **git status -s** → shows the m in green
36. **touch .gitignore** -Files that you don't want tag
37. **touch mylog.log** -Example of log file write anything
38. **in gitignore write mylogs.log to ignoire all the file with this name**
39. **run git status -**it will not show that file added in the gitignore
40. **add this in gitignore /mylogs.log -**if you want to add the same file in same folder
41. **git rm –cached logs/mylogs.log** -to remove from the stages area

42. ***.log in gitignore – To add file with .log**
43. **ignore a folder foldername/**
44. **Git commit -a -m “Ignoring the files in the ignore files”**

Creating a Branch by command line

1. **git status** -Explain about the on branch master
2. **git branch -**it shows all the branch in the git
3. **git branch update1** -New branch with name update1 is created
4. **git branch** – now two branches will be shown
5. **git checkout update1** -Switch to the branch update1

Using the merge feature in the git command line

Make some changes in the code you have written in any of the file then use next command

1. **git add -A**
2. **git commit -m “Fixed the programs and added more features”**
3. **git status**
4. **git checkout master** – To move to the master Branch
5. **git checkout update1**
6. **add the comment in the file any comment**
7. **git commit -a -m “added some comments”**
8. **git status**
9. **git checkout master** -for merging the branch to master then we will switch to master
10. **git log**
11. **git merge update1** -this merge the update code to master
12. **git log -p -2** -We can see the logs of the same commit in update to the master now.
13. **git checkout -b nodeintegration** -It will switch and make the branch in this command

Create a js file and add some printing code in it then commit

1. **git commit -a -m “Added the node file in the code”**
2. **git status**
3. **when we switch to master the file of node will be lost now**
4. **git checkout master**
5. **in about.html write anything**
6. **git commit -a -m “modified the about page”**
7. **git status**
8. **git checkout nodeintegration**

9. **files again comes**

Now talk about GitHub service (Microsoft service)

On command line

1- **git checkout master**
2- **remote repository**
3- **paste the repository origin URL in the GitHub**
4- **git remote**
5- **git remote -v (push and fetch urls)**
6- **git push origin master (as the private repo then it will show there is no repo)**
7- **create new folder on desktop on clone**
8- **copy the clone url**
9- **git clone url foldernanme (clone to the folder name)**