1. Set your username and email in git config

```
>> git config –global user.name “sumana”
>> git config –global user.email “sumana.gurajapali@gmail.com”
```

2. Create a new branch named "feature-branch" and switch to it.

```
>> git checkout -b feature-branch
```

3. List all branches in the repository.

```
 >> git branch -a
```

4. Delete the branch “feature-branch”

```
>> git checkout main
>> git branch -d feature-branch
```

5. How do you undo the last commit

```
>> git revert HEAD --no-edit
```

6. Create a new branch names “conflict-branch”

```
>> git checkout -b conflict-branch
```

7. Create a another branch named “feature1”

```
>> git checnkout -b feature1
```

8. Make some changes in to feature1 branch

```
>> echo “Some changes” >> file.txt
>> git add file.txt
>> git commit -m “Made some changes in feature1”
```

9.Merge “feature1” branch into main branch

```
>> git checkout main
 >> git merge feature1
```

10.Make changes in “conflict-branch”, in the same file and line that you had made changes in feature1

```
>> git checkout conflict-branch
>> echo “Conflicting changes” > file.txt
>> git add file.txt
>> git commit -m “Made conflicting changes in conflict-branch”
```

11.Merge master into conflict-branch [Attach screenshot of terminal & file]

```
>> git merge main
```

12.Resolve merge conflicts

```
>> git add file.txt
>> git commit -m “Resolved merge conflicts”
```

13.Add a remote named "origin" pointing to a GitHub repository.

```
>> git remote add origin https://github.com/sumana/todo app.git
```

14.Fork a repository on GitHub and clone it to your local machine.

```
>> git clone https://github.com/sumana/forked-repository.git
```

15.Create a new branch on your fork, make changes, and open a pull request to the original repository.

```
>> git checkout -b new-feature
>> git add .
>> git commit -m “Added new feature”
>> git push origin new-feature
```

16.Comment on a PR and suggest improvements

In `feature.txt`, the variable name `x` is not descriptive.consider using a more meaninful name, such as `userInput`:

```
- let x = getInput();
+ let userInput = getInput();
```

17.Create a Git alias for the command `git log --oneline` named `gitlol`.

```
>>git config –global alias.gitolol “log –oneline”
```

18.Create a pre-commit hook

```
>>echo ‘#!/bin/sh’ > .git/hooks/pre-commit
>>echo ‘echo “Running pre-commit hook...”’ >> .git/hooks/pre-commit
>>chomd +x .git/hooks/pre-commit
```

19.You have made local changes in your branch, but you need to switch to another branch urgently without committing. How would you handle this situation?

```
>>git stash
>>git checkout another-branch
>>git stash pop
```

20.You accidentally deleted a file in your local repository. How do you restore it using Git?

```
>>Git checkout -- path/to/deleted-file
```

21,You have committed changes to your branch but forgot to include a file. How do you add the file to the last commit without creating a new commit?

```
>>git add forgot-file.txt
>>git commit –amend --no-edit
```

22,You want to discard all changes in your working directory and revert to the last commit. What Git command would you use?

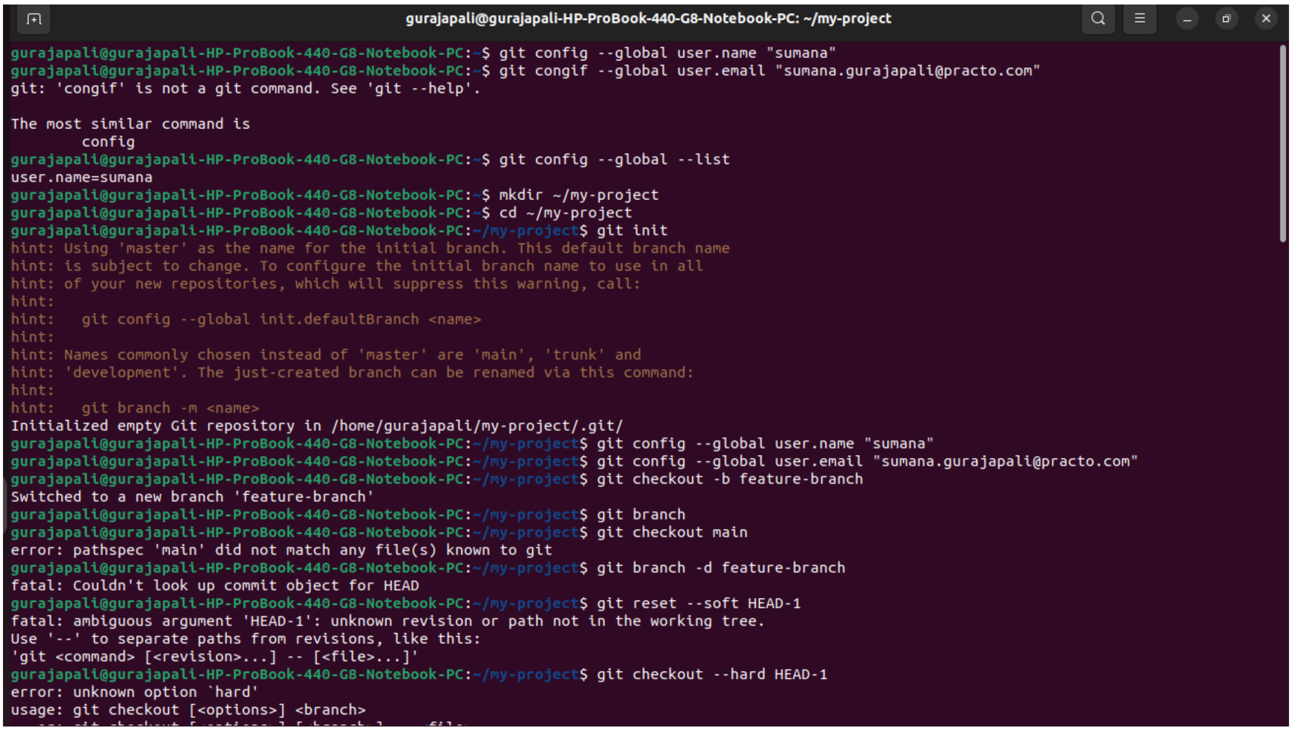
```
>>git reset --hard
```

23.You need to view a specific commit's changes. What Git command can be used to show the changes introduced by a particular commit?

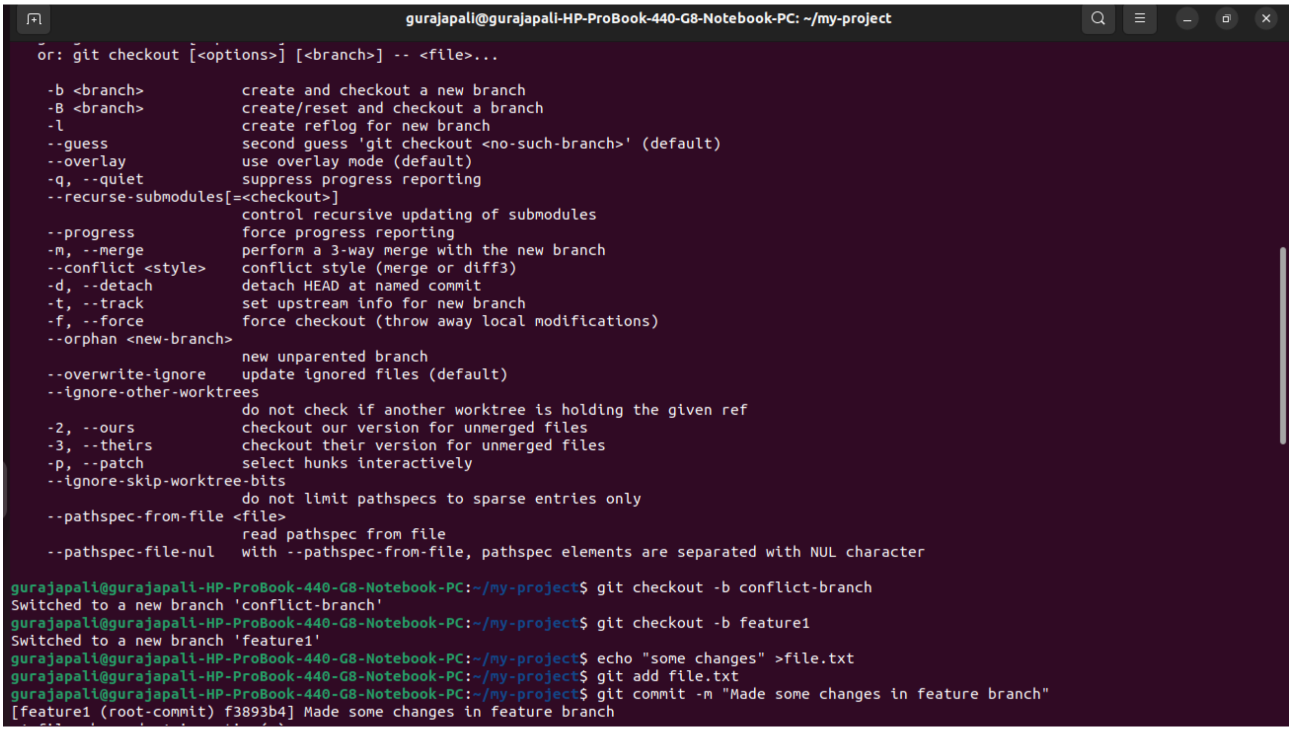
```
>>git show <commit-hash>
```

24.You want to change a commit message, after you have already committed, how do you do so?

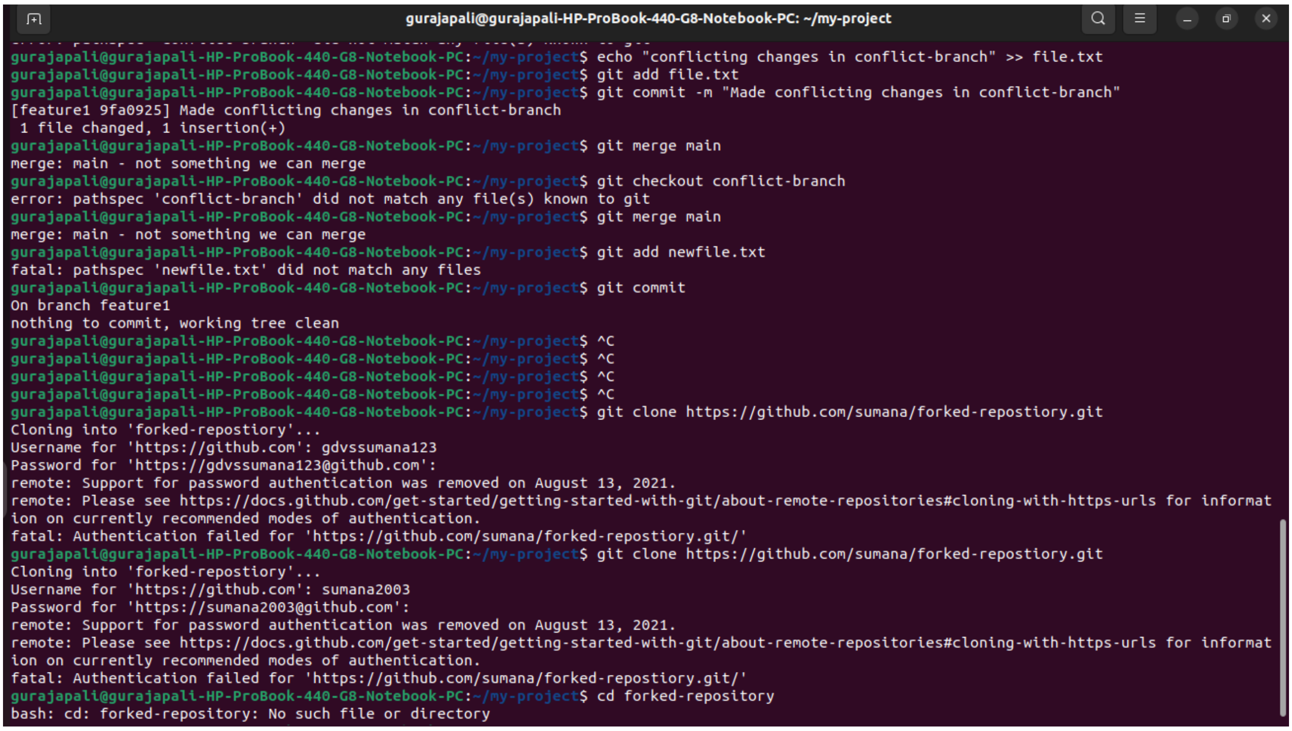
```
>>git commit –amend -m “New commit message”
```


gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC: ~/my-project

```
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~$ git config --global user.name "sumana"
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~$ git congif --global user.email "sumana.gurajapali@practo.com"
git: 'congif' is not a git command. See 'git --help'.

The most similar command is
        config
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~$ git config --global --list
user.name=sumana
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~$ mkdir ~/my-project
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~$ cd ~/my-project
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/gurajapali/my-project/.git/
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git config --global user.name "sumana"
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git config --global user.email "sumana.gurajapali@practo.com"
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git branch
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git checkout main
error: pathspec 'main' did not match any file(s) known to git
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git branch -d feature-branch
fatal: Couldn't look up commit object for HEAD
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git reset --soft HEAD-1
fatal: ambiguous argument 'HEAD-1': unknown revision or path not in the working tree.
Use '--' to separate paths from revisions, like this:
'git <command> [<revision>...] -- [<file>...]'
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git checkout --hard HEAD-1
error: unknown option `hard'
usage: git checkout [<options>] <branch>
```


gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC: ~/my-project

```
   or: git checkout [<options>] [<branch>] -- <file>...

    -b <branch>           create and checkout a new branch
    -B <branch>           create/reset and checkout a branch
    -l                    create reflog for new branch
    --guess               second guess 'git checkout <no-such-branch>' (default)
    --overlay             use overlay mode (default)
    -q, --quiet           suppress progress reporting
    --recurse-submodules[=<checkout>]
                          control recursive updating of submodules
    --progress            force progress reporting
    -m, --merge           perform a 3-way merge with the new branch
    --conflict <style>    conflict style (merge or diff3)
    -d, --detach          detach HEAD at named commit
    -t, --track           set upstream info for new branch
    -f, --force           force checkout (throw away local modifications)
    --orphan <new-branch>
                          new unparented branch
    --overwrite-ignore    update ignored files (default)
    --ignore-other-worktrees
                          do not check if another worktree is holding the given ref
    -2, --ours            checkout our version for unmerged files
    -3, --theirs          checkout their version for unmerged files
    -p, --patch           select hunks interactively
    --ignore-skip-worktree-bits
                          do not limit pathspecs to sparse entries only
    --pathspec-from-file <file>
                          read pathspec from file
    --pathspec-file-nul   with --pathspec-from-file, pathspec elements are separated with NUL character

gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git checkout -b conflict-branch
Switched to a new branch 'conflict-branch'
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git checkout -b feature1
Switched to a new branch 'feature1'
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ echo "some changes" >file.txt
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git add file.txt
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git commit -m "Made some changes in feature branch"
[feature1 (root-commit) f3893b4] Made some changes in feature branch
```


gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC: ~/my-project

```
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ echo "conflicting changes in conflict-branch" >> file.txt
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git add file.txt
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git commit -m "Made conflicting changes in conflict-branch"
[feature1 9fa0925] Made conflicting changes in conflict-branch
 1 file changed, 1 insertion(+)
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git merge main
merge: main - not something we can merge
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git checkout conflict-branch
error: pathspec 'conflict-branch' did not match any file(s) known to git
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git merge main
merge: main - not something we can merge
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git add newfile.txt
fatal: pathspec 'newfile.txt' did not match any files
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git commit
On branch feature1
nothing to commit, working tree clean
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ ^C
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ ^C
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ ^C
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ ^C
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git clone https://github.com/sumana/forked-repostiory.git
Cloning into 'forked-repostiory'...
Username for 'https://github.com': gdvssumana123
Password for 'https://gdvssumana123@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for informat
ion on currently recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/sumana/forked-repostiory.git/'
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ git clone https://github.com/sumana/forked-repostiory.git
Cloning into 'forked-repostiory'...
Username for 'https://github.com': sumana2003
Password for 'https://sumana2003@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/get-started/getting-started-with-git/about-remote-repositories#cloning-with-https-urls for informat
ion on currently recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/sumana/forked-repostiory.git/'
gurajapali@gurajapali-HP-ProBook-440-G8-Notebook-PC:~/my-project$ cd forked-repository
bash: cd: forked-repository: No such file or directory
```

25.You accidentally deleted a file in your local repository. How do you restore it using Git?

>>Git add wow.text

>>git commit –amed –no-edit

26.You have committed changes to your branch but forgot to include a file. How do you add the file to the last commit without creating a new commit?

>>Git add wow.text

>>git commit –amed –no-edit

27.You want to discard all changes in your working directory and revert to the last commit. What Git command would you use?

>>Git reset –hard HEAD

28.You need to view a specific commit's changes. What Git command can be used to show the changes introduced by a particular commit?

>>git show commit-hash

29.You want to change a commit message, after you have already committed, how do you do so?

>>Git commit –amend

30.Your colleague has made changes in their branch, and you want to incorporate those changes into your branch without merging. How do you achieve this?

>>Git cherry-pick commit-hash

31.You've made several commits on a branch, but you want to club them into a single commit before pushing to the remote repository. How would you do that?

>>Git rebase -i HEAD~N

32.You accidentally staged a file that you don't want to commit. How do you unstage it?

```
>>Git reset HEAD file
```

33.You don’t want to commit files that have .yml in the end, and also files inside folder config. How do you do that?

```
>>Git add .gitignore(in gitignore file add *.yml)
```

34.You want to see a list of all the files changed in the last commit. What Git command would you use?

```
>>Git diff –name-only HEAD
```

35.You realize that your local branch is outdated, and you want to fetch the latest changes from the remote repository. How do you do this without merging?

```
>>Git fetch
```

36.You accidentally deleted a branch. How do you recover it?

```
>>Git checkout -b deleted-branch-name commit-hash
```

37.You want to remove untracked files and directories from your working directory. What Git command would you use?

```
>>Git clean -n
```

38.You have a commit from a feature branch that you want to apply to the main branch without merging the entire feature branch.

```
>>Git checkout main
>>Git cherry-pick<commit-hash>
```

39.You mistakenly committed a change to the wrong branch and need to apply that commit to the correct branch.

```
>>Git checkout correct-branch
>>Git che        rry-pick <commit-hash>
>>Git checkout wrong-branch
>>Git revert<commit-hash>
```

40.There is a series of commits on a feature branch, but you only want to cherry-pick a specific range of commits.

```
>>Git checkout main
```

41.You want to clone a GitHub repository onto your local machine, but you only need a specific branch. How can you achieve this?

```
>>Git clone -b <branch name> <repo-url>
```

42.You've made changes to your local repository and want to push them to your fork on GitHub. What Git commands would you use?

```
>>Git add.
>>Git commit -m “message”
>>Git push origin branch name
```

43.You want to create a new branch both locally and on GitHub to work on a new feature. What commands would you use?

```
>>Git checkout -b new
>>Git add .
```

```
>>Git commit -m “msg”
>>Git push
```

44.You want to see the commit history of a GitHub repository. How can you do this using Git commands?

```
>>Git log
```

45.You've accidentally committed sensitive information and want to remove the commit from both your local and remote repositories on GitHub. What commands would you use?

```
>>Git reset id
>>Git revert <commit-hash>
```

46.You want to delete a remote branch on GitHub. What Git command would you use?

```
>>Git push origin –delete<branch-name>
```

47.Create a git repository for all your assignments and upload them in it. Ask your peers to code review it, and you need to code review your peers assignments

48.Create a pull request on any open source library on github, attach the pull request link to the readme file of this project’s repository