**-Veera Naresh IT**

# 1). How can you copy a particular commit made in one branch to another?

1. Git merge
2. git rebase
3. git cherry-pick
4. git stash

# 2. One of your colleagues accidentally deleted a branch, and has already pushed the changes to the central git repo. There are no other git repos, and none of your other teammates had a local copy. How would you recover this branch?

1. check out the latest commit to this branch and then check it out as a new branch.
2. Check out the previous commit to this branch and then check it out as a new branch.
3. we can recover by using git rebase
4. clone the entire repository into local from existing branch

# How can you view the commit history in Git?

1. git logs
2. git history
3. git commits
4. git log

# What is a merge conflict?

1. An error in the Git configuration
2. A disagreement among team members
3. Overlapping changes between branches
4. An issue with remote repository synchronization

# How do you fetch the latest updates from a remote repository without merging them?

1. git pull
2. git get
3. git fetch
4. git update

# How can you undo the most recent commit?

1. git revert HEAD
2. git reset HEAD~1
3. git undo
4. git back

# What does the git cherry-pick command do?

1. Picks a random commit from history
2. Applies changes from a specific commit to the current branch
3. Deletes a specific commit
4. Merges two unrelated branches

# Which file can you configure to ensure that certain files are never committed to the local Git repository?

1. Ignore.git
2. .gitignore
3. gitignore.txt
4. git.ignore

# how to resolve a conflict in git?

1. Identify the files that have caused the conflict.
2. Make the necessary changes in the files so that conflict does not arise again. 3.Add these files by the command git add. And commit

4.all of the above

# 10 Your colleague unable to merge the changes, getting conflict issues Cause of conflicts in git?

1. A ‘conflict’ arises when the commit that has to be merged has some change in same file.
2. A ‘conflict’ arises when the commit that has to be merged has some change in one place
3. A conflict arises without pull
4. A conflict arises if two developers work on the different branches.

# User made changes to multiple files and want to commit them separately. Which Git command should you use to stage only one of the changed files?

1. git add .
2. git cherry-pick
3. git add file\_name
4. git commit -m "Commit message"

# I have done some changes like added files from working directory to staging area now I want to switch to another branch is it possible now?

Yes No

# User unable to push the changes into git remote repository so what would be the reasons?

1. Multiple users working with same branch
2. user need to pull the changes before push
3. User working with individual branches
4. 1 and 2

# 14. Which is the good practice to create branch?

1. create from main branch where we configured integrations.
2. create from development branch only to protect our main branch
3. we can create from any branch
4. we should not create branches every time

# 15. Naresh created one branch and added some updates from working directory to staging area now Naresh wants to create one more branch what he has to do?

1. Stash the changes first and create new branch
2. Undo the changes from staging to working directory and now create new branch 3.First Commit changes and push to remote repository then create new branch 4.All of the above

# 16. Multiple developers working with same repository so which is best practice to work with git?

1. Each developer work on individual branches
2. All developers work with same branch only
3. pull before push
4. 1 and 3

# 17. Difference between git rebase and git merge

1. One additional merge commit will create while doing git rebase
2. Git rebase is linear operation it will not create any additional commit for merge
3. Git merge will create one more additional commit while merge time.
4. 2 and 3

# 18. How to revert a bad commit that is already pushed?

1. A new commit can be created that reverts changes done in the bad commit. It can be done using git revert <name of bad commit>
2. git reset head~1
3. A new commit can be created that reverts changes done in the bad commit. It can be done using git revert <name of bad commit>
4. Not possible already pushed

# 19.What is the HEAD in GIT?

1.A HEAD is a reference to the currently checked out commit

2. It is a symbolic reference to the branch that we have checked out 3.it refers git hub remote repository latest commit

* 1. and 2

# 20. How do we put a local repository on GitHub server?(repository not existing in remote)

1. git init, git add git commit git remote add, git push
2. git init, git add git commit git remote -v, git push
3. git push
4. git clone and push

# 21. How can you take copy of the code one repo to other repo

1. git clone 2.git fork 3.git merge

4. all of the above

# You want to discard all changes in the working directory and revert to the state of the last commit. Which Git command should you use?

1. git reset --hard HEAD
2. git checkout -- .
3. git clean -f
4. git revert HEAD

# You want to view the differences between the current working directory and the staging area. Which Git command should you use?

1. git diff
2. git status
3. git log
4. git show

# You want to create a new branch named "feature" and switch to it. Which Git command should you use?

1. git checkout -b feature
2. git branch feature
3. git branch -b feature
4. git switch -b feature

# What is the purpose of the ".gitignore" file in a Git repository?

1. To ignore all files in the repository
2. To specify files that should be tracked by Git
3. To specify files and directories that should be ignored by Git
4. To force Git to commit all changes, regardless of file status

# My changes have been added to the staging area, but I realized I don't want to commit them at this time. What process can I apply now?

1. Git Ignore
2. Git Stash
3. Git restore <filename>
4. All of the above
5. **I have pushed my changes to the remote repository. How can I delete a particular commit from both the remote and local repositories?**
6. Git reset Head~1
7. Git revert
8. Git restore <filename>
9. Git Stash
10. **My changes have been added and committed to the local repository, but I realized I don't want to push them at this time. So, I ran git reset --hard <previous commit>. What happens to the recent commit changes?**
11. Files will be Deleted for present commit in both local and working directory
12. Files will be Deleted for present commit in local repo only
13. Files will be there in staging area for particular commit and local repo
14. Files will be there in working area for particular commit but deleted in local
15. **My changes have been added and committed to the local repository, but I realized I don't want to push them at this time. So, I ran git reset --mixed <previous commit>. What happens to the recent commit changes?**
16. Files will be Deleted for recent commit in both local and working directory
17. Files will be Deleted for recent commit in local repo only
18. Files will be there in staging area for particular recent and local repo
19. Files will be there in working area for recent commit but deleted in local repo.
20. **User ran git revert for a particular commit. After that, the user ran again git pull and pushed again. In this case, will the file deleted by using revert still exist, or is it permanently deleted?**
21. Exists
22. Deleted.
23. Exists in Working directory
24. None of the above

**31. Which command applies the latest stashed changes and removes them from the stash list?**

A) git stash apply

B) git stash pop

C) git stash drop

D) git stash save

**32. What is the difference between git stash apply and git stash pop?**

A) Both are the same commands

B) git stash apply applies and removes the stash, while git stash pop only applies the stash

C) git stash apply applies without removing the stash, while git stash pop applies and removes the stash.

D) git stash apply only applies specific files, while git stash pop applies all changes

**33. How do you apply a specific “3” stash from the stash list?**

a) git stash apply

b) git stash apply stash@{3}.

c) git stash pop stash@{3}

d) b and c

**34. My changes have been added to the staging area, but I realized these changes should be added to a different branch. Is it possible to check out to that branch?**

a) run git stash and checkout only

b) commit the changes and checkout

c) not possible to checkout if any changes are pending at staging area

d) All of the above.

**35. Which command is used to resolve a conflict after a merge?**

a) git conflict resolve

b) git resolve

c) git commit --resolve

d) Conflicts are resolved manually by editing files.

**36. After resolving a conflict, what is the next step to finalize the merge?**

a) Run git merge --finish

b) Run git add and commit to save the resolved changes and push.

c) Run git stash to store the resolved changes

d) Run git push to update the remote

**37. What is a git pull fast-forward?**

a) A merge that requires manual conflict resolution

b) A merge that moves the branch pointer without creating a new commit.

c) A merge that automatically creates a new commit

d) A merge that resets the branch history

**38. Which command would you use to avoid a fast-forward merge and force a merge commit?**

a) git merge --force

b) git merge --no-ff.

c) git rebase --fast

d) git merge –abort

**39. The user forgot to run git pull before committing their changes, and now they are unable to push the changes. The user ran git pull --rebase to resolve the issue**

a) fetches the changes from the remote repository and rebases the local commits on top of the fetched changes with out creating new commit.

b) fetches the changes from the remote repository and rebases the local commits on top of the fetched changes with creating new commit

C) fetches the changes from the remote repository to only staging are

D) All of the above

**40. When would you prefer using git merge instead of git rebase?**

a) When you want a clean and linear commit history

b) When you want to preserve all branch history and merge commits.

c) When you want to delete old commits

d) When you want to reapply commits to a different branch