# Git Test Sheet

## Part 1: Git Basics

1. 1. What does `git init` do?

* Creates a new branch.
* Initializes a new repository.
* Adds a remote repository.
* Starts a commit.

1. 2. Which command stages files for commit?

* git commit
* git add
* git push
* git log

1. 3. What does the `.gitignore` file do?

* Tracks deleted files.
* Lists files and directories to be ignored by Git.
* Removes files from the repository.
* Tracks staged changes.

1. 4. What is a Git repository?

* A server that hosts projects.
* A local directory with Git version control.
* A list of changes.
* A remote branch.

1. 5. How do you check the current Git version?

* git --version
* git version
* git info
* git status --version

1. 6. Which file contains the repository metadata?

* metadata.json
* .git/ folder
* config.yaml
* .gitignore

1. 7. What is the default branch in Git (historically)?

* main
* develop
* master
* origin

1. 8. How do you see the status of your repository?

* git log
* git branch
* git status
* git fetch

1. 9. What does `git clone` do?

* Creates a new branch locally.
* Copies an existing repository to a new directory.
* Synchronizes branches.
* Deletes a remote repository.

1. 10. What does `git rm` do?

* Stages files for removal.
* Removes files from the working directory and staging area.
* Deletes branches.
* Removes untracked files.

## Part 2: Git Commits

1. 11. What does `git commit` do?

* Saves changes in the working directory.
* Records changes to the repository.
* Deletes files.
* Pushes changes to a remote.

1. 12. What does `git commit -m` do?

* Commits all changes without staging.
* Shows staged changes.
* Commits changes with a message.
* Creates a branch.

1. 13. How do you see commit history?

* git branch
* git status
* git log
* git fetch

1. 14. How do you undo the last commit without losing changes?

* git revert HEAD
* git reset HEAD~1 --soft
* git stash
* git rm

1. 15. Which command amends the last commit?

* git commit --edit
* git commit --amend
* git commit --force
* git commit --fix

1. 16. What is a commit hash?

* A short description of the commit.
* A unique identifier for each commit.
* A reference to a branch.
* A summary of remote changes.

1. 17. What does `HEAD` represent in Git?

* The first commit in a branch.
* The most recent commit.
* The currently checked-out branch.
* The name of the repository.

1. 18. How do you discard unstaged changes?

* git discard
* git reset --hard
* git clean
* git stash

1. 19. What does `git diff` show?

* Differences between commits.
* Staged changes.
* Differences between the working directory and staging area.
* All of the above.

1. 20. What is a commit message used for?

* To track branches.
* To describe changes in a commit.
* To list conflicts.
* To create a backup of files.

## Part 3: Branching and Merging

1. 21. How do you create a new branch?

* git new branch <branch-name>
* git branch <branch-name>
* git checkout -b <branch-name>
* Both B and C

1. 22. How do you delete a branch?

* git delete <branch>
* git branch -d <branch>
* git remove <branch>
* git rm branch <branch>

1. 23. Which command merges two branches?

* git combine
* git merge
* git pull
* git rebase

1. 24. What does a merge conflict indicate?

* The repository is corrupted.
* There are overlapping changes in branches.
* The remote branch is out of sync.
* The repository has untracked files.

1. 25. What does `git branch` do?

* Lists all branches in the repository.
* Deletes a branch.
* Pushes changes to a branch.
* Creates a new branch.

1. 26. How do you switch branches in Git?

* git branch
* git checkout <branch-name>
* git switch <branch-name>
* Both B and C

1. 27. How do you rebase a branch?

* git merge <branch>
* git rebase <branch>
* git pull --rebase
* Both B and C

1. 28. What is a fast-forward merge?

* Combines all commits into one.
* Moves the HEAD pointer to the target branch.
* Deletes a branch after merging.
* Resolves merge conflicts.

1. 29. How do you rename a branch?

* git rename branch
* git branch -m <new-name>
* git update branch <new-name>
* git branch --rename <new-name>

1. 30. What does `git stash` do?

* Saves changes temporarily.
* Deletes changes.
* Stages changes for commit.
* Reverts changes.

## Part 4: Remote Repositories

1. 31. What does `git remote` do?

* Lists remote repositories.
* Adds a new branch remotely.
* Deletes a remote repository.
* Fetches remote branches.

1. 32. Which command pushes changes to the remote repository?

* git add
* git commit
* git push
* git remote

1. 33. How do you pull changes from the remote repository?

* git fetch
* git pull
* git sync
* git merge origin

1. 34. How do you clone a remote repository?

* git copy <url>
* git fetch <url>
* git clone <url>
* git pull <url>

1. 35. What does `git fetch` do?

* Downloads changes without merging.
* Merges changes from remote.
* Pushes changes to the remote repository.
* Deletes remote branches.

1. 36. How do you add a remote repository?

* git remote add <name> <url>
* git add remote <name>
* git set origin <url>
* git remote --add <name>

1. 37. What does `origin` refer to in Git?

* A local branch.
* The default remote repository name.
* The initial commit.
* A backup branch.

1. 38. How do you see remote branches?

* git branch -r
* git show branches
* git remote -v
* git log

1. 39. How do you delete a remote branch?

* git branch -d <branch>
* git branch --remote delete
* git push origin --delete <branch>
* git remote remove <branch>

1. 40. What does `git pull` do?

* Downloads changes and merges them into the current branch.
* Pushes changes to the remote branch.
* Deletes a branch.
* Stages all changes.

## Part 5: Advanced Git

1. 41. What does `git rebase` do?
2. 42. How do you view tags in a repository?
3. 43. How do you create an annotated tag?
4. 44. What is the difference between `git fetch` and `git pull`?
5. 45. What does `git cherry-pick` do?
6. 46. How do you resolve merge conflicts?
7. 47. How do you track changes in Git using logs?
8. 48. What does `git reset` do?
9. 49. How do you perform a hard reset?
10. 50. What is the purpose of Git hooks?