1. What is Git, and how does it differ from other version control systems like SVN or Mercurial?

- **Git** is a distributed version control system that tracks code changes. It lets multiple developers collaborate without relying on a single server, unlike **SVN**. **Mercurial** is also distributed but varies in performance and workflow.

2. How to install Git on Ubuntu?

- Run: sudo apt-get install git
- Verify with: git --version

3. What is a Git repository, and how to create one?

- A **Git repository** tracks file changes.
    - Initialize: git init
    - Clone: git clone <repository_url>

4. Difference between a local and remote repository?

- **Local**: Stored on your computer for development.
- **Remote**: Hosted on a server for collaboration (e.g., GitHub).

5. Check current Git configuration settings

- View all: git config --list
- Check specific: git config user.name or git config user.email

6. Purpose of the .gitignore file

- Specifies files/directories for Git to ignore, like build files or secrets.

7. Basic Git Workflow

- Create/clone repo, modify files, stage (git add), commit (git commit), push (git push).

8. Tracking new files

- git add .: Stages all changes
- git add <file>: Stages specific file

9. What git commit does

- Saves staged changes. Use clear, concise messages (e.g., "Add feature X").

10. Check repository status

- git status: Shows untracked/modified files and current branch

11. View changes before committing

- git diff: Shows changes in modified files

12. Commit message options

- git commit -m "message": Inline message
- git commit: Opens editor for detailed message

13. What is a Git branch?

- Parallel version of the repo for independent work. Essential for collaboration.

14. Create/switch branches

- Create: git branch <branch_name>
- Switch: git checkout <branch_name>

15. What is git merge?

- Combines branches. Resolving conflicts may be necessary if changes overlap.

16. What is git rebase?

- Repositions your branch at another's tip, rewriting history (unlike git merge).

17. Delete branches

- Locally: git branch -d <branch_name>
- Remotely: git push origin --delete <branch_name>

18. Use git stash

- Saves uncommitted changes. Use git stash apply to retrieve them.

19. Clone a repository

- git clone <repository_url>

20. git pull vs. git fetch

- git pull: Fetches and merges
- git fetch: Only downloads updates

21. Push changes

- git push: Sends commits to the remote

22. What are remotes?

- References to remote repos. Add: git remote add <name> <url>

23. Rename/remove remote

- Rename: git remote rename <old_name> <new_name>
- Remove: git remote remove <name>

24. What is a pull request?

- Request to merge changes, often used for code reviews.

25. View commit history

- git log

26. Revert to previous commit

- View: git checkout <commit>
- Reset: git reset
- Undo: git revert

27. git reset vs. git checkout vs. git revert

- git reset: Changes history
- git checkout: Switches branches/views commits
- git revert: Creates new commit to undo changes

28. Find changes to a file

- git blame <file>

29. Git tags

- Mark specific commits (e.g., releases). Create: git tag <tag_name>

30. Handling large files

- Use **Git LFS** for efficient storage and tracking.