1.What are the advantages of using Git?

A. One of the biggest advantages of Git is its branching capabilities. Unlike centralized version **control** systems, Git branches are cheap and easy to merge. This facilitates the feature branch workflow popular with many Git users. Feature branches provide an isolated **environment** for every change to your codebase.

2. What language is used in Git?

A. C,pearl,Python etc.

3. What is the meaning of “Index” or “Staging Area” in Git?

A. The **Git index** is used as a **staging area** between your working directory and your repository. You can use the **index** to build up a set of changes that you want to commit together. When you create a commit, what is committed is what is currently in the **index**, not what is in your working directory.

4. What is the process for creating a repository in Git?

A.1) In the upper-right corner of any page, use the drop-down menu, and select **New repository**.

2)Type a short, memorable name for your **repository**. ...

3)Optionally, add a description of your **repository**. ...

4)Choose to **make** the **repository** either public or private. ...

5)Select Initialize this **repository** with a README.

6)Click **Create repository**.

5. What is ‘head’ in Git and how many heads can be created in a repository?

A. By default, there is a **head** in every **repository** called master. A **repository can** contain **any** number of **heads**. At **any** given time, one **head** is selected as the “current **head**.” This **head** is aliased to **HEAD**, always in capitals".

6. Why do we need branching in Git?

A. **Branching** allows each developer to **branch** out from the original code base and isolate their work from others. It also helps **Git** to easily **merge** versions later on.

7. Write a way to create a new branch in Git?

A.1.git pull

2.git checkout -b branchname

3.git push origin brachname

8. How do you define a ‘conflict’ in Git?

**A.Git** can handle most merges on its own with automatic merging features. A **conflict** arises when two separate branches have made edits to the same line in a file, or when a file has been deleted in one branch but edited in the other. **Conflicts** will most likely happen when working in a team environment.

9. How to resolve a conflict in Git?

A. The most direct way to resolve a merge conflict is to edit the conflicted file. Open the conflicted file in your favorite editor. Resolve conflicts by removing all the conflict dividers and fixing conflicting code.

Then use **git add conflictfile** to stage the new merged content.

Finalize merge by creating a new commit – git commit -m “Resolving merge conflict”

Finally push the commit.

10. What is the function of ‘git config’?

A. The git config command is a convenience function that is used to set Git configuration values on a global or local project level. These configuration levels correspond to . gitconfig **text** files. Executing git config will modify a configuration **text** file.

11. What is Git fork?

A. A **fork** is a copy of a repository. Forking a repository allows you to freely experiment with changes without affecting the original project. Most commonly, **forks** are used to either propose changes to someone else's project or to use someone else's project as a starting point for your own idea.

12. Difference between fork, branch and clone?

A.A **fork** is really a Github (not Git) construct to store a **clone** of the repo in your user account. As a **clone**, it will contain all the **branches in the** main repo at the time you made the **fork**. ... These commits are pulled from either my **fork** or my **branch** to the main repo. A commit is a set of changes to the code.

13. What's the difference between a "pull request" and a "branch"?

A. A **branch** is just a separate version of the code. A **pull request** is when someone take the repo, makes their own **branch**, does some changes, then tries to merge that **branch in**.

14. What is the difference between "git pull" and "git fetch"?

A. **git fetch** is the command that tells your local git to retrieve the latest meta-data info from the original (yet doesn't do any file transfering. It's more like just checking to see if there are any changes available). **git pull** on the other hand does that AND brings (copy) those changes from the remote repository.

15. How to revert previous commit in Git?

A. If you want to **revert** the **last commit** just do **git revert** <unwanted **commit** hash> ; then you can push this new **commit**, which undid your **previous commit**. To fix the detached head do **git** checkout <current branch> .

16. Explain the advantages of Forking Workflow?

A. The main **advantage** of the **Forking Workflow** is that contributions can be integrated without the need for everybody to push to a single central repository. Developers push to their own server-side repositories, and only the project maintainer can push to the official repository.

17. Difference between HEAD, working tree and index, in Git?

A. **Working trees**: They are nothing but the files that you are currently **working** on. **HEAD**: **HEAD** is a pointer to the branch or commit that you last checked out, and which will be the parent of a new commit if you make it. ... **Index**: The **git** "**index**" is where you place files you want commit to the **git** repository.

18. How to identify if a certain branch has been merged into master?

A. Run git branch --merged master command and verify if the branch is in the response.

19. What is the use of a Git clone?

A. **Usage**. **git clone** is primarily **used** to point to an existing repo and make a **clone** or copy of that repo at in a new directory, at another location. The original **repository** can be located on the local filesystem or on remote machine accessible supported protocols. The **git clone** command copies an existing **Git repository**.

20. What is Git stash?

A. The **git stash** command takes your uncommitted changes (both staged and unstaged), saves them away for later use, and then reverts them from your working copy.

21. When should I use "git stash"?

A. Use git stash when you want to record the current state of the working directory and the index.

22. What is Git stash drop?

A. **Git stash** is a temporary storage. When you're ready to continue where you left off, you can restore the saved state easily

23. What is Git stash save?

A. When you **Git stash** or **Git stash save**, **Git** will actually create a **Git** commit object with some name and then **save** it in your repo. So it means that you can view the list of stashes you made at any time. You can see the list of stashes made. And the most recent **stash** made is in the top.

24. What README.MD ? What is its purpose? What does MD stands for?

A. . **md** is markdown . **README**.**md** is used to generate **the** html summary you see at **the** bottom of projects. **Github** has **their** own flavor of Markdown. Order of Preference: If you have two files named **README** and **README**.**md** , **the** file named **README**.**md** is preferred, and it will be used to generate **github's** html summary.

25. How to create repository from command prompt?

1. Create a directory to contain the project.

2.Go into the new directory.

3.Type git init.

4.create ReadMe.MD file

5. git commit -m "first commit"

6.git remote add origin https://github.com/sushmithamedi/new\_repo.git

7.git push -u origin master

26. What is the function of ‘git checkout’ in Git?

A. The **git checkout** command lets you navigate between the branches created by **git** branch . Checking out a branch updates the files in the working directory to match the version stored in that branch, and it tells **Git** to record all new commits on that branch.

27. How can you bring a new feature in the main branch?

A. Create a **new**-**branch** Use a separate **branch** for each **feature** or issue you work on. After creating a **branch**, check it out locally so that any changes you make will be on that **branch**. This checks out a **branch** called **new**-**feature** based on master , and the -b flag tells Git to create the **branch** if it doesn't already exist.

28. What is the function of ‘git rm’?

A. The git rm command can be used to remove individual files or a collection of files.

29. What is the function of ‘git stash apply’?

A. **git stash** temporarily shelves (or stashes) changes you've made to your working copy so you can work on something else, and then come back and re-**apply** them later on.

30. What is the use of ‘git log’?

A. **Git logs** allow you to review and read a history of everything that happens to a repository.

31. What is ‘git add’ is used for?

A. **git add** is the first command in a chain of operations that directs **Git** to "save" a snapshot of the current project state, into the commit history. When **used** on its own, **git add** will promote pending changes from the working directory to the staging area.

32. What is 'git diff' is used for?

A. **git diff** is a multi-use **Git** command that when executed runs a **diff** function on **Git** data sources. These data sources can be commits, branches, files and more.

33. What is ‘git status’ is used for?

A. The **git status** command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven't, and which files aren't being tracked by **Git**. **Status** output does not show you any information regarding the committed project history.

34. Can we create multiple branch with one command?

A. No, it’s not possible with a single command.However we can write a script with loop stmt.

35. what is the command that is used to delete a branch?

A.local branch: git branch -d branchname

Remote branch: git push origin --delete branchname

36. What is another option for merging in git?

A. Rebase as an **Alternative** to **Merge**. While **merging** is definitely the easiest and most common way to integrate changes, it's not the only one: "Rebase" is an **alternative** means of integration.

37. How to remove a file from git without removing it from your file system?

A. command: git rm --cached filename

38. Use of  "git rebase" instead of "git merge"?

A. But, **instead** of using a merge commit, rebasing re-writes the project history by creating brand new commits for each commit in the original branch. The major benefit of rebasing is that you get a much cleaner project history. First, it eliminates the unnecessary merge commits required by **git merge** .

39. What is a repository in Git?

A. A Git repository is the .git/ folder inside a project. This repository tracks all changes made to files in your project, building a history over time.

40. Command used to write a commit message?

A. **command** : git **commit** -m

41. What does commit object contain?

A. The **commit object contains** the directory tree **object** hash, parent **commit** hash, author, committer, date and message.

42. Write one use-case of Github?

A. **GitHub** is **used** to manage the collaborative development of projects in corporate world.

Eg: banking websites,ecommerce websites like amazon.etc,.

43. Name some alternative of Git?

A. GitHub, SVN (Subversion), Bitbucket, Perforce, and **Mercurial** are the most popular alternatives and competitors to Git.

44. What is a gist in Git?

A. All **gists** are **git** repositories, so they are automatically versioned, forkable and usable as a **git** repository. A **Gist** is a snippet of code hosted by Github that has all of the benefits of a Github repository, but provides them to you in a more lightweight way.

45. What is a gist programming?

A. **Gist**: Is an additional feature added to github to allow the sharing of code snippets, notes, to do lists and more.

46. Name any two Git repository hosting services which are common?

A.Gitlab and BitBucket.