GIT BRANCH AND MERGE

git branch

The git branch command is used to list, create, or delete branches in a Git repository. Here are some common use cases:

List Branches:

Example(in git bash):

```
git branch
```

This command lists all the branches in the repository. The currently active branch is highlighted with an asterisk (*).

Create a New Branch:

Example(in git bash):

```
git branch <branch_name>
```

This command creates a new branch with the specified name. However, it does not switch to the new branch. To switch to the new branch, you need to use git checkout or git switch (in Git versions 2.23 and later).

Delete a Branch:

Example(in git bash):

```
git branch -d <branch_name>
```

This command deletes the specified branch. The -d option stands for "delete." If the branch contains changes that are not merged, you may need to force delete with -D (capital D).

git merge

The git merge command is used to combine changes from different branches. The most common scenario is merging a feature branch into the main branch (e.g., merging a feature branch into master).

Switch to the Target Branch:

Before merging, ensure you are on the branch where you want to merge changes. For example, to merge a feature branch into master:

Example(in git bash):

```
git checkout master
```

Merge the Branch:

Example(in git bash):

```
git merge <branch_name>
```

This command incorporates changes from the specified branch into the current branch.

Resolve Merge Conflicts:

If Git encounters conflicts during the merge (changes in both branches that cannot be automatically merged), it marks the conflicts, and you need to resolve them manually. After resolving conflicts, you complete the merge with:

Example(in git bash):

```
git merge --continue
```

or cancel the merge with:

Example(in git bash):

```
git merge --abort
```

GIT PUSH ANG GIT PULL

git push

The git push command is used to upload local changes to a remote repository. It is commonly used to share your local changes with others or to update a remote repository with your latest work.

Basic Push:

Example(in git bash):

```
git push origin <branch_name>
```

This command pushes the commits from the specified local branch to the corresponding branch on the remote repository (origin is the default name for the remote repository).

Force Push:

In some cases, you may need to force push (overwrite remote changes with your local changes). Be cautious with force pushes, especially in a collaborative environment.

Example(in git bash):

```
git push -f origin <branch_name>
```

git pull

The git pull command is used to fetch and merge changes from a remote repository into the current branch. It is a combination of git fetch and git merge.

Basic Pull:

Example(in git bash):

```
git pull origin <branch_name>
```

This command fetches changes from the remote repository (origin) and merges them into the current local branch.

Pull with Rebase:

Instead of merging, you can use rebase to incorporate changes from the remote repository. This results in a cleaner history but should be used with caution, especially when working with shared branches.

Example(in git bash):

```
git pull --rebase origin <branch_name>
```

These commands are fundamental for collaboration in Git. git branch and git merge help manage branches and integrate changes, while git push and git pull facilitate the sharing of code between local and remote repositories. Understanding and mastering these commands are essential for effective version control and collaboration in Git.