PROGRAM 5

Fetch the latest changes from a remote repository and rebase your local branch onto the updated remote branch.

Understanding the Commands

Let's break down each command step-by-step:

STEP 1: cd repository name:

• Purpose: This command changes the current directory to the specified repository.

- Explanation:

- cd: Stands for "change directory."

- repository name: Replace this with the actual name of your Git repository.

• Example: cd my-project This would change your current working directory to the folder named "my-project".

STEP 2: git fetch origin:

- Purpose: This command fetches the latest changes from the remoterepository named "origin" to your local repository.

- Explanation:

- git fetch: Fetches commits, files, and other objects from a remote repository.

- origin: Refers to the default remote repository, usually set up when you clone a repository.

• What happens:

- It downloads new commits, branches, and tags from the remoterepository.

- These changes are stored locally but are not immediately merged into your current branch.

STEP 3: git rebase origin/main:

- Purpose: This command reapplies your local commits on top of the latestcommits from the remote "main" branch.
- Explanation:
- git rebase: Rebases your current branch onto another branch.
- origin/main: Refers to the remote "main" branch.
- What happens:
- It replays your local commits on top of the latest commits from the remote "main" branch.
- This creates a linear commit history, making it easier to review and understand the changes.

STEP 4: git log:

- Purpose: This command displays a log of commits in your localrepository.
- **Explanation:**
- git log: Shows the commit history.
- What happens:
- It displays a list of commits, including their hash, author, date, and commit message.
- You can use various options with git log to customize the output, such as filtering by author, date, or commit message. Key Points to Remember:
- Always fetch before rebasing to ensure you have the latest changesfrom the remote repository.

*Rebasing can be a powerful tool, but it can also be complex. Use it with caution, especially in shared repositories.

- Git log is a valuable tool for understanding the history of your project and identifying specific changes.

By understanding these commands and their functions, you can effectively manage your Git workflow and keep your local repository upto-date with the remote repository.