Git Assignment Questions

1. Setting Up Git:

Q1: Install Git on your system and configure your name and email using the following commands:

- `git config global user.name "Your Name"`

- `git config global user.email "your.email@example.com"`

Q2: How would you verify that Git has been installed and properly configured? Provide the command and the expected output.

- Command: `git version` and `git config global list`

- Expected Output:

- `git version x.x.x` (Git version number).

- Config list showing the set name and email (e.g., `user.name=Your Name`, `user.email=your.email@example.com`).

Q3: Initialize a new Git repository in an empty directory on your computer using

`git init`.

2. Basic Git Operations:

Q4: Create a new text file named `hello.txt` in your repository. Add some content to it. Then, stage the file for commit using the `git add` command.

Q5: Commit the changes you made to the `hello.txt` file with a meaningful commit message. Provide the Git command to commit and the expected output.

- Command: `git commit -m "Added hello.txt with initial content"`

- Expected Output:

- `[main (root-commit) xxxxxx] Added hello.txt with initial content`

- `1 file changed, 1 insertion(+)`

Q6: After committing your changes, use the `git status` command to check the state of your repository. Explain the output.

- Output:

- If no changes are made: `On branch main, nothing to commit, working tree clean.`

- If there are untracked changes: Lists untracked files.

Q7: How can you view the commit history of a repository? Use the `git log` command and describe what information it provides.

- Information: Commit hash, author name, date, and commit message.

3. Branching and Merging:

Q8: What is the purpose of branching in Git? How do branches help in software development?

- Purpose: To work on different features/versions in parallel without affecting the main code.

Q9: Create a new branch called `feature-branch` and switch to it using the appropriate Git command.

- Command: `git checkout -b feature-branch`

Q10: Create a new file named `feature.txt` on your new branch and commit the changes. Then, switch back to the `main` branch.

Q11: Merge the `feature-branch` into the `main` branch. What command would you use to merge the changes, and what happens if there are no conflicts?

- Command: `git merge feature-branch`

- Outcome: If no conflicts, changes are merged automatically, and the commit history is updated.

Q12: What is a merge conflict? Create a scenario where a merge conflict occurs and explain how you would resolve it.

- Merge Conflict: Occurs when changes in the same file/lines are made in different branches.

- Resolution: Use a code editor to manually fix conflicts, then stage the file and commit.

4. Working with Remote Repositories:

Q13: What is a remote repository in Git? How is it different from a local repository?

- Remote Repository: Hosted on a server for collaboration. Local is on your machine.

Q14: Clone a remote repository from GitHub to your local machine using the `git clone` command. Provide the URL of a public repository to clone.

- Example Command: `git clone https://github.com/example/repo.git`

Q15: After cloning the repository, make a small change (e.g., edit `README.md`), and commit the changes to your local repository.

Q16: Push your local commits to the remote repository. What Git command is used to push changes to a remote repository? Explain how you would use it.

- Command: `git push origin main`

- Pushes changes in the `main` branch to the remote repository.

Q17: Fetch the latest changes from the remote repository using the `git fetch` command. What is the difference between `git fetch` and `git pull`?

- Difference:

- `git fetch`: Downloads changes without merging.

- `git pull`: Fetches and merges changes.

5. Undoing Changes in Git:

Q18: After making several commits, you realize that a commit message needs to be changed. How can you edit the last commit message using Git?

- Command: `git commit amend -m "Updated commit message"`

- Outcome: Changes the message of the last commit, but keeps the changes intact.