**GIT ASSIGNMENT**

**Question 3**

Step 1: Create a feature branch.

Navigate to the directory of your Git repository.

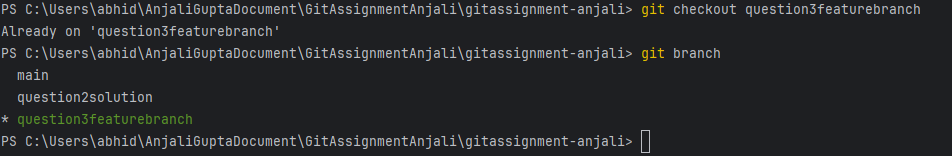
Run the following command to create a new feature branch:

Syntax : git checkout -b feature-branch



Step 2: Switch to the new branch.

Syntax : git checkout feature-branch



Step 3: open the file and make some changes to it.

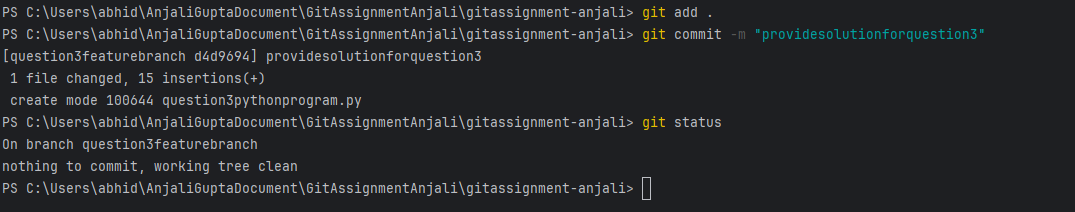
A screen shot of a computer program

Description automatically generated

Step 4: Add and commit the changes to the new branch.

Syntax : git add .

git commit -m "Your commit message" .



Step 5: Push the changes to the new feature branch.

Once i've committed my changes to the feature branch, i push them to the remote repository using the following command:

Syntax : git push origin feature-branch

A computer screen shot of a program

Description automatically generated

Step 6: Create a pull request.

Go to the GitHub repository in your web browser.

Switch to the "Pull Requests" tab And Click "New pull request" button.

A screenshot of a computer

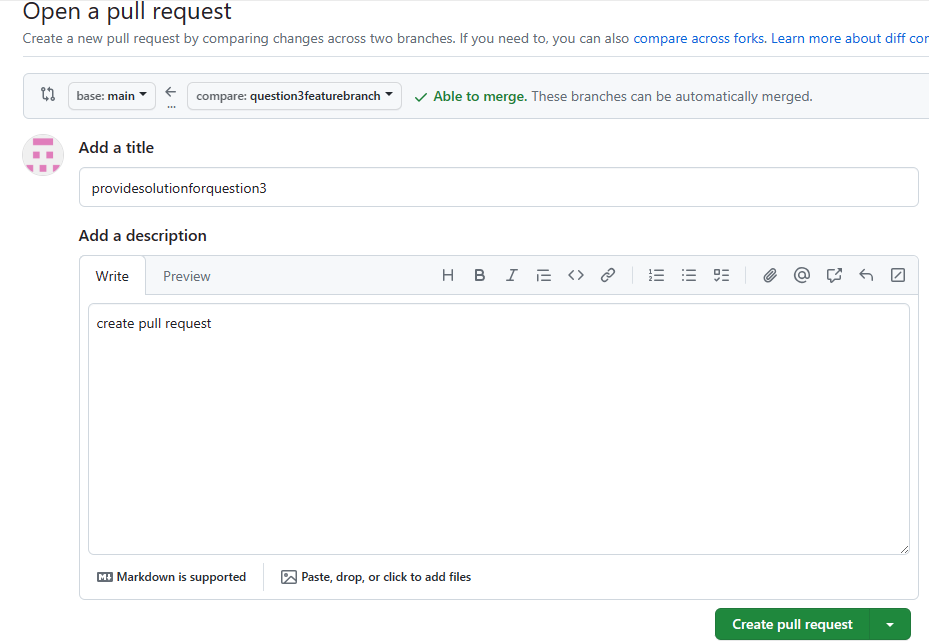
Description automatically generated

Select the base branch (usually "main") and the compare branch (your feature branch).

A screenshot of a computer

Description automatically generated

Review the changes and click on the "Create pull request" button to create the pull request.

A screenshot of a computer

Description automatically generated

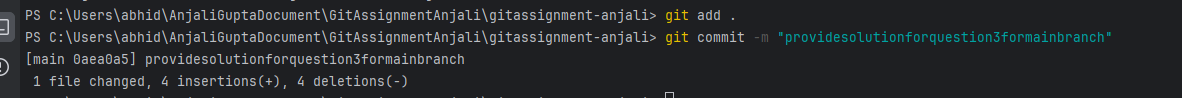
Step 6: As another user in the master branch make some changes to the same file.

Another user (or me on main branch) makes changes to the same file in the main branch.

A screenshot of a computer program

Description automatically generated

Step 7: Add and commit the changes to the main branch.



Step 8: Push the changes to the master branch.

A screen shot of a computer program

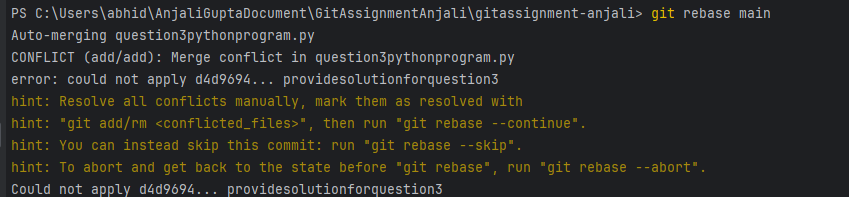
Description automatically generated

Note: There will be a conflict in the pull request, how do we resolve it??

A screenshot of a computer

Description automatically generated

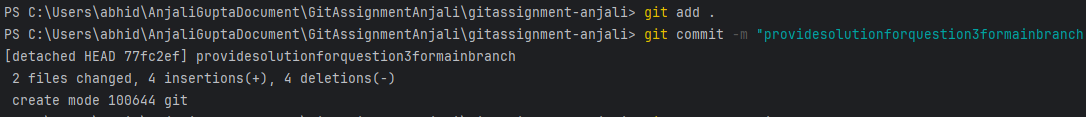
Step 9: Resolve the conflict in the pull request of feature branch.



A screenshot of a computer

Description automatically generated

Hint: git rebase



Type command ‘ git mergetool’ for seeing changes and take action according to file and resolve conflict

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Question 4**

Step 1: Step 1: Create a feature branch.

Step 2: Switch to the new branch.

open the file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

Step 3: Identify the commit or commits that you want to "cherry-pick"(Note the hash of the commit or commits that you want to "cherry-pick")

Step 4: Use the "git checkout" command to switch to the branch where you want to apply the changes.

Step 5: Use the "git cherry-pick" command followed by the commit hash(es) that you want to apply.

**Question 5**

Step 1: Step 1: Create a feature branch.

Step 2: Switch to the new branch.

open the file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

open the same file and make some changes to it.

Add and commit the changes to the new branch.

Step 3: Use the "git log" command to view the commit history and identify the commit to which you want to reset.

Step 4: Use the "git reset" command followed by the desired reset type and the commit hash

Step 5: Verify that the reset was successful by using the "git log" command again.

Step 6: Use the "git log" command to view the commit history and identify the commit that you want to reverse.

Step 7: Use the "git revert" command followed by the commit hash or reference to which you want to revert. (Hint: git revert <commit hash>)

Step 8: Verify that the revert was successful by using the "git log" command again.

Note: Identify the difference between git log after git reset and git r evert.