**Explain how to resolve the conflict during merge**

**Steps and Commands**

**1. Verify if master is in a clean state**

bash

git checkout master

git status

**2. Create a new branch GitWork**

bash

git branch GitWork

git checkout GitWork

**3. Add a file hello.xml**

bash

echo "<message>Hello from GitWork branch</message>" > hello.xml

**4. Update the content of hello.xml**

bash

echo "<message>Updated in GitWork branch</message>" > hello.xml

git status

**5. Commit the changes to the branch**

bash

git add hello.xml

git commit -m "Added hello.xml in GitWork branch"

**6. Switch back to master**

bash

git checkout master

**7. Add a hello.xml file in master with different content**

bash

echo "<message>Hello from Master branch</message>" > hello.xml

git add hello.xml

git commit -m "Added hello.xml in master branch"

**8. Observe the log (graph view)**

bash

git log --oneline --graph --decorate --all

**9. Check differences between master and GitWork**

bash

git diff master GitWork

**10. View differences visually with P4Merge**

bash

git mergetool

Ensure P4Merge is set up as Git's merge tool.

**11. Merge GitWork into master (will cause conflict)**

bash

git merge GitWork

**12. Observe Git conflict markers in hello.xml**

bash

cat hello.xml

You will see:

php-template

<<<<<<< HEAD

<message>Hello from Master branch</message>

=======

<message>Updated in GitWork branch</message>

>>>>>>> GitWork

**13. Use 3-way merge tool to resolve the conflict**

bash

git mergetool

Choose the correct content, save, and close.

**14. Commit the resolved conflict**

bash

git add hello.xml

git commit -m "Resolved merge conflict in hello.xml"

**15. Check status and ignore backup merge files**

bash

git status

echo "\*.orig" >> .gitignore

**16. Commit the .gitignore changes**

bash

git add .gitignore

git commit -m "Added .gitignore to exclude merge backup files"

**17. List all branches**

bash

git branch -a

**18. Delete the merged branch**

bash

git branch -d GitWork

**19. View the log after merge**

bash

git log --oneline --graph --decorate