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

When a merge conflict occurs in Git, it means that changes made in two different branches affect the same part of a file, and Git cannot automatically decide which version to keep. To resolve the conflict, you first identify the files with conflicts using git status. Open the conflicted files in an editor, where Git marks the conflicting sections with <<<<<<<, =======, and >>>>>>> to show the different changes. Manually edit the file to keep the correct changes or combine them as needed, then remove the conflict markers. Once resolved, save the file, run git add <filename> to stage the changes, and finally commit with git commit to complete the merge. This process ensures that both branches’ changes are integrated in a way that fits your project’s requirements.

**Implement conflict resolution when multiple users are updating the trunk (or master) in such a way that it results into a conflict with the branch’s modification.**

**Step-by-Step Instructions**

**Step 1 – Verify Master is in Clean State**

git checkout master

git status

Ensure there are no uncommitted changes.

**Step 2 – Create Branch "GitWork" and Add File**

git branch GitWork

git checkout GitWork

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

git add hello.xml

**Step 3 – Update hello.xml and Observe Status**

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

git status

**Step 4 – Commit Changes in Branch**

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

**Step 5 – Switch to Master**

git checkout master

**Step 6 – Add a Different Version of hello.xml to Master**

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

git add hello.xml

**Step 7 – Commit Changes in Master**

git commit -m "Add hello.xml in master branch with different content"

**Step 8 – View Log**

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

**Step 9 – Check Differences (Command Line)**

git diff master GitWork

**Step 10 – View Differences in P4Merge**

git mergetool

Ensure P4Merge is configured as the default merge tool.

**Step 11 – Attempt to Merge Branch into Master**

git merge GitWork

A **merge conflict** will appear.

**Step 12 – Observe Git Markup**

* Open hello.xml in your editor.

<<<<<<< HEAD

Content from master

=======

Content from GitWork branch

>>>>>>> GitWork

**Step 13 – Use 3-Way Merge Tool (P4Merge) to Resolve**

git mergetool

Manually choose or combine content to resolve the conflict.

**Step 14 – Commit Resolved Changes**

git add hello.xml

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

**Step 15 – Add Backup File to .gitignore**

echo "\*.orig" >> .gitignore

git add .gitignore

git commit -m "Add .orig files to .gitignore"

**Step 16 – List All Branches**

git branch

**Step 17 – Delete Merged Branch**

git branch -d GitWork

**Step 18 – View Final Log**

git log --oneline --graph --decorate







