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

git checkout master

git status

**2. Create a branch “GitWork” and add a file hello.xml**

git branch GitWork

git checkout GitWork

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

**3. Update the content of hello.xml and observe the status**

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

git status

**4. Commit the changes to reflect in the branch**

git add hello.xml

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

**5. Switch to master**

git checkout master

**6. Add a file hello.xml to master with different content**

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

**7. Commit the changes to master**

git add hello.xml

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

**8. Observe the log**

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

**9. Check the differences using Git diff tool**

git diff master GitWork

**10. Use P4Merge tool to list out visual differences between master and branch**

git mergetool --tool=p4merge

**11. Merge the branch into master**

git merge GitWork

**12. Observe the Git mark-up (conflict indicators)**

cat hello.xml

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

git mergetool

**14. Commit the changes to master after resolving the conflict**

git add hello.xml

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

**15. Observe the Git status and add backup file to .gitignore**

git status

echo "\*.orig" >> .gitignore

**16. Commit the changes to .gitignore**

git add .gitignore

git commit -m "Added .gitignore to ignore backup files"

**17. List out all available branches**

git branch -a

**18. Delete the branch merged into master**

git branch -d GitWork

**19. Observe the log**

git log --oneline --graph --decorate