Exercise 4 : GIT-HOL

# Steps

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

git status

## 2. Create a branch 'GitWork' and add a file 'hello.xml'

git checkout -b GitWork  
echo "<message>Hello from GitWork branch</message>" > hello.xml

## 3. Update the content of 'hello.xml' and observe the status

echo "<update>Branch modification</update>" >> hello.xml  
git status

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

git add hello.xml  
git commit -m "Added hello.xml and updated 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 with different content"

## 8. Observe the log

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

## 9. Check the differences with Git diff tool

git diff GitWork

## 10. Use P4Merge for visual comparison

git config --global merge.tool p4merge  
git config --global mergetool.p4merge.cmd "\"C:/Program Files/Perforce/p4merge.exe\" \"$BASE\" \"$LOCAL\" \"$REMOTE\" \"$MERGED\""  
git difftool master GitWork

## 11. Merge the branch into master

git merge GitWork

## 12. Observe the Git markup (conflict markers)

cat hello.xml

## 13. Use 3-way merge tool to resolve conflict

git mergetool

## 14. Commit the resolved file

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

## 15. Check status and ignore backup files

git status  
echo "\*.orig" >> .gitignore

## 16. Commit the .gitignore

git add .gitignore  
git commit -m "Added .orig to .gitignore"

## 17. List all branches

git branch

## 18. Delete merged branch

git branch -d GitWork

## 19. View final log

git log --oneline --graph --decorate