**Please follow the instructions to complete the hands-on. Each instruction expect a command for the Git Bash.**

1. Verify if master is in clean state.
2. Create a branch **“GitWork”.** Add a file “hello.xml”.
3. Update the content of “hello.xml” and observe the status
4. Commit the changes to reflect in the branch
5. Switch to master.
6. Add a file **“hello.xml”** to the master and add some different content than previous.
7. Commit the changes to the master
8. Observe the log by executing **“git log –oneline –graph –decorate –all”**
9. Check the differences with Git diff tool
10. For better visualization, use P4Merge tool to list out all the differences between master and branch
11. Merge the bran to the master
12. Observe the git mark up.
13. Use 3-way merge tool to resolve the conflict
14. Commit the changes to the master, once done with conflict
15. Observe the git status and add backup file to the .gitignore file.
16. Commit the changes to the .gitignore
17. List out all the available branches
18. Delete the branch, which merge to master.
19. Observe the log by executing **“git log –oneline –graph –decorate”**

**COMMANDS:**

git checkout master

git status

git checkout -b GitWork

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

git add hello.xml

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

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

git status

git add hello.xml

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

git checkout master

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

git add hello.xml

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

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

git diff GitWork

git merge GitWork

git add hello.xml

git commit -m "Resolved conflict between master and GitWork in hello.xml"

git status

echo "\*.orig" >> .gitignore

git add .gitignore

git commit -m "Ignore merge backup files"

git branch -a

git branch -d GitWork

git log --oneline --graph --decorate



