**4. Git-HOL Hands On Assignment Notes**

**Process of Resolving a Merge Conflict**

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

git checkout master

git status.

### 2. Create a branch ****GitWork****

git checkout -b GitWork

### 3. Add a file hello.xml

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

### 4. Update content and observe status

git status

### 5. Commit changes to branch

git add hello.xml

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

### 6. Switch to master

git checkout master

### 7. Add file hello.xml in master with different content

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

git add hello.xml

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

### 8. Observe log

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

### 9. Check differences

git diff master GitWork

### 10. Merge branch to master

git merge GitWork

We will get a **merge conflict** on hello.xml.

### 11. Observe Git mark up

Open hello.xml in an editor we will see:

<<<<<<< HEAD

<greeting>Hello from Master branch</greeting>

=======

<greeting>Hello from GitWork branch</greeting>

>>>>>>> GitWork

### 12. Resolve conflict with 3-way merge tool

echo "<greeting>Hello from BOTH Master and GitWork</greeting>" > hello.xml

or use:

git mergetool

### 13. Commit resolved changes

git add hello.xml

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

### 14. Observe status & add backup files to .gitignore file

git status

echo "\*.orig" >> .gitignore

### 15. Commit .gitignore file

git add .gitignore

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

### 16. List all branches

git branch

### 17. Delete merged branch

git branch -d GitWork

### 18. Observe log again

git log --oneline --graph --decorate