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

git status  
  
Output:  
On branch master  
nothing to commit, working tree clean

### 2. Create a branch “GitWork”. Add a file “hello.xml”.

git checkout -b GitWork  
echo '<message>Hello from GitWork branch</message>' > hello.xml  
git add 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  
  
Output:  
Changes not staged for commit:  
 modified: hello.xml

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

git add hello.xml  
git commit -m 'Update hello.xml in GitWork branch'  
  
Output:  
[GitWork abc1234] Update hello.xml in GitWork branch

### 5. Switch to master.

git checkout master

### 6. Add a file “hello.xml” to the master and add some different content than previous.

echo '<message>Hello from master branch</message>' > hello.xml  
git add hello.xml

### 7. Commit the changes to the master

git commit -m 'Add hello.xml in master branch'  
  
Output:  
[master def5678] Add hello.xml in master branch

### 8. Observe the log

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

### 9. Check the differences with Git diff tool

git diff master GitWork

### 10. Use P4Merge tool to list all differences

p4merge hello.xml hello.xml  
  
Output:  
[Opens P4Merge GUI showing side-by-side file differences]

### 11. Merge the branch to the master

git merge GitWork  
  
Output:  
Auto-merging hello.xml  
CONFLICT (content): Merge conflict in hello.xml

### 12. Observe the git markup.

cat hello.xml  
  
Output:  
<<<<<<< HEAD  
<message>Hello from master branch</message>  
=======  
<message>Updated content in GitWork branch</message>  
>>>>>>> GitWork

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

p4merge  
  
[Resolve by editing file to final content]

### 14. Commit the changes to the master, once done with conflict

git add hello.xml  
git commit -m 'Resolve merge conflict in hello.xml'

### 15. Observe the git status and add backup file to the .gitignore file.

git status  
echo '\*.bak' >> .gitignore  
git add .gitignore

### 16. Commit the changes to the .gitignore

git commit -m 'Add .gitignore for backup files'

### 17. List out all the available branches

git branch  
  
Output:  
\* master  
 GitWork

### 18. Delete the branch which merged to master.

git branch -d GitWork  
  
Output:  
Deleted branch GitWork (was abc1234).

### 19. Observe the log

git log --oneline --graph --decorate