Name:- Vidya CS TRACK:JAVA FSE

1. Verify if master is in clean state

```
git checkout master

git status
```

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

```
git checkout -b GitWork

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

git status
```

3. Update the content of “hello.xml” and observe the status

```
echo "<message>Hello updated in GitWork</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 the master and add some different content than previous

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

git status
```

7. Commit the changes to the master

```
git add hello.xml

git commit -m "Added hello.xml in master branch"
```

8. Observe the log by executing:

```
git log --oneline --graph --decorate –all
```

9. Check the differences with Git diff tool

```
git diff GitWork master -- hello.xml
```

10. For better visualization, use P4Merge tool

```
git difftool GitWork master -- hello.xml
```

11. Merge the branch to the master

```
git merge GitWork
```

12. Observe the git mark up (conflict markers if any)

```
cat hello.xml
```

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

```
git mergetool
```

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

```
git add hello.xml

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

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

```
git status

echo "*.orig" >> .gitignore
```

16. Commit the changes to the .gitignore

```
git add .gitignore

git commit -m "Added .gitignore rule for backup files"
```

17. List out all the available branches

```
git branch
```

18. Delete the branch which merged to master

```
git branch -d GitWork
```

19. Observe the log again

```
git log --oneline --graph --decorate
```