**Lab Exercise 3– Git Merge and Merge Conflict**

**Objective**

To learn how to merge branches in Git and resolve conflicts when two branches modify the same part of a file.

**Prerequisites**

* Git installed on your machine (git --version)
* Basic knowledge of Git repository and branching

**Steps**

**1. Create a New Repository**

mkdir git-merge-lab

cd git-merge-lab

git init

**2. Create an Initial File and Commit**

echo "Line 1: Hello Git" > notes.txt

echo "Line 2: This is the main branch" >> notes.txt

git add notes.txt

git commit -m "Initial commit with notes.txt"

**3. Create and Modify in a Feature Branch**

git checkout -b feature-branch

echo "Line 3: Added from feature branch" >> notes.txt

git add notes.txt

git commit -m "Updated notes.txt in feature branch"

**4. Modify the Same File in Main Branch**

git checkout main

echo "Line 3: Added from main branch" >> notes.txt

git add notes.txt

git commit -m "Updated notes.txt in main branch"

Now both branches (main and feature-branch) have changes on **Line 3**.

**5. Attempt to Merge and Create Conflict**

git merge feature-branch

Git will show a **merge conflict** because both branches modified the same line.

**6. Inspect the Conflict**

Open notes.txt, you’ll see something like this:

Line 1: Hello Git

Line 2: This is the main branch

<<<<<<< HEAD

Line 3: Added from main branch

=======

Line 3: Added from feature branch

>>>>>>> feature-branch

* <<<<<<< HEAD → content from current branch (main)
* ======= → separator
* >>>>>>> feature-branch → content from merging branch

**7. Resolve the Conflict**

Edit notes.txt manually, keeping the desired content, e.g.:

Line 1: Hello Git

Line 2: This is the main branch

Line 3: Added from both main and feature branch

**8. Mark Conflict as Resolved and Commit**

git add notes.txt

git commit -m "Resolved merge conflict between main and feature-branch"