# Git-HOL-Solutions4

## Objective

Learn how to resolve merge conflicts in Git when multiple users make changes that result in file conflicts.

## Prerequisites

1. Git installed and configured (username, email, preferred editor).  
2. P4Merge installed and configured for visual diff/merge:  
 git config --global merge.tool p4merge  
 git config --global mergetool.p4merge.path "C:/Program Files/Perforce/p4merge.exe"  
3. Remote repository ready (GitHub or GitLab).  
4. Ensure working directory is clean (no pending changes).

## Step 1: Verify master is clean

git checkout master  
git status # Ensure no uncommitted changes

## Step 2: Create branch and make changes

# Create and switch to branch  
git checkout -b GitWork  
  
# Add file and commit  
echo "<msg>Hello from branch</msg>" > hello.xml  
git add hello.xml  
git commit -m "Add hello.xml in GitWork"  
  
# Modify file  
echo "<msg>Updated from GitWork branch</msg>" > hello.xml  
git status  
git commit -am "Update hello.xml in GitWork"

## Step 3: Create conflicting changes on master

# Switch to master  
git checkout master  
  
# Add conflicting file content  
echo "<msg>Hello from master branch</msg>" > hello.xml  
git add hello.xml  
git commit -m "Add hello.xml in master with different content"

## Step 4: View differences

# CLI diff  
git diff GitWork  
  
# Visual diff with P4Merge  
git mergetool

## Step 5: Merge and resolve conflict

# Attempt merge (will cause conflict)  
git merge GitWork  
  
# Git marks conflict in hello.xml with <<<<<<<, =======, >>>>>>> markers  
  
# Resolve conflict manually or use 3-way merge tool:  
git mergetool  
  
# After resolving:  
git add hello.xml  
git commit -m "Resolve merge conflict between master and GitWork"

## Step 6: Update .gitignore for backup files

# Ignore backup files from merge tool  
echo "\*.orig" >> .gitignore  
git add .gitignore  
git commit -m "Add merge backup files to .gitignore"

## Step 7: Cleanup and verify

# List branches  
git branch -a  
  
# Delete merged branch  
git branch -d GitWork  
  
# View log  
git log --oneline --graph --decorate

## Verification Checklist

✓ Conflict detected during merge  
✓ Conflict resolved and committed  
✓ Backup files ignored via .gitignore  
✓ Branch deleted after successful merge