EDWIN A

GIT ASSIGNMENT

Question 1

Step 1: Create a new Git repository.

Step 2: Create a file and commit changes.

```
Admin@Monica MINGW64 ~/question1 (master)
$ touch file1.txt
```

```
Admin@Monica MINGW64 ~/question1 (master)
$ echo "hello this is my task1" > file1.txt

Admin@Monica MINGW64 ~/question1 (master)
$ cat file1.txt
hello this is my task1

Admin@Monica MINGW64 ~/question1 (master)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

Admin@Monica MINGW64 ~/question1 (master)
$ git commit -m "this is v1"
[master (root-commit) 709ad83] this is v1
 1 file changed, 1 insertion(+)
```

Step 3: View the commit history of your repository.

```
Admin@Monica MINGW64 ~/question1 (master)
$ git log
commit 709ad8348a50797b575518ff9664f3444ece2bed (HEAD -> master)
Author: aredwin <edwinise2025@gmail.com>
Date:   Tue May 20 15:41:40 2025 +0530

    this is v1
```

Step 4: Open the file you created earlier and make some changes to it.

```
Admin@Monica MINGW64 ~/question1 (master)
$ vi file1.txt

Admin@Monica MINGW64 ~/question1 (master)
$ cat file1.txt
hello this is my task1

hi im edwin
this task must be completed by today
```

Step 5: Check the file you modified is now marked as "modified" and unstaged.

Hint (git status)

```
Admin@Monica MINGW64 ~/question1 (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   file1.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

Step 6: Stage the changes you made to the file and commit the changes to the repository.

```
Admin@Monica MINGW64 ~/question1 (master)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

Admin@Monica MINGW64 ~/question1 (master)
$ git commit -m "modified the changes"
[master 31e9c2a] modified the changes
 1 file changed, 3 insertions(+)
```

```
Admin@Monica MINGW64 ~/question1 (master)
$ git remote add origin https://github.com/aredwin/git-assignment.git

Admin@Monica MINGW64 ~/question1 (master)
$ git remote set-url origin https://github.com/aredwin/git-assignment.git
```

Step 7: Clone the repository you have created in GitHub.

```
Admin@Monica MINGW64 ~ (master)
$ git clone https://github.com/aredwin/git-assignment.git
Cloning into 'git-assignment'...
remote: Enumerating objects: 9, done.
remote: Counting objects: 100% (9/9), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 9 (delta 0), reused 6 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (9/9), done.
```

Step 8: Fetch the changes, navigate into the cloned repository using the command line, and use the command git fetch to fetch any changes that have been made to the original repository since you cloned it.

```
Admin@Monica MINGW64 ~ (master)
$ cd git-assignment

Admin@Monica MINGW64 ~/git-assignment (main)
$ git fetch
```

Step 9: Pull changes, merge the changes you just fetched into your local copy of the repository, and use the command git pull.

```
Admin@Monica MINGW64 ~/git-assignment (main)
$ git pull
Already up to date.
```

Question 2

Step 1: Clone the repository you have created in GitHub.

Step 2: Create a new branch using the command.

Step 3: Switch to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (main)
$ git checkout local
Switched to branch 'local'
```

Step 4: Make some changes to the code in your local copy of the repository.

```
Admin@Monica MINGW64 ~/git-assignment (local)
$ vi file1.txt
```

Step 5: Commit changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (local)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

Admin@Monica MINGW64 ~/git-assignment (local)
$ git commit -m "added the content in the new branch"
[local 60c1378] added the content in the new branch
 1 file changed, 1 insertion(+)
 create mode 100644 file1.txt
```

Step 6: Switch back to the original branch

```
Admin@Monica MINGW64 ~/git-assignment (local)
$ git checkout master
branch 'master' set up to track 'origin/master'.
Switched to a new branch 'master'
```

Step 7: Merge the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git merge local
fatal: refusing to merge unrelated histories
```

Step 8: Push changes to the original branch

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git push origin master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 721 bytes | 360.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/aredwin/git-assignment.git
   31e9c2a..7015bd5  master -> master
```

Question 3

Step 1: Create a feature branch.

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git branch feature
```

Step 2: Switch to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git checkout feature
Switched to branch 'feature'
```

Step 3: open the file and make some changes to it.

```
Admin@Monica MINGW64 ~/git-assignment (feature)
$ echo "this is my 3rd task" > file1.txt

Admin@Monica MINGW64 ~/git-assignment (feature)
$ cat file1.txt
this is my 3rd task
```

Step 4: Add and commit the changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

Admin@Monica MINGW64 ~/git-assignment (feature)
$ git commit -m "modified the file for 3rd task"
[feature 3be60eb] modified the file for 3rd task
 1 file changed, 1 insertion(+), 8 deletions(-)
```

Step 5: Push the changes to the new feature branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature)
$ git push origin feature
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 305 bytes | 152.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:      https://github.com/aredwin/git-assignment/pull/new/feature
remote:
To https://github.com/aredwin/git-assignment.git
 * [new branch]      feature -> feature
```

Step 6: Create a pull request.

Step 6: As another user in the master branch make some changes to the same file.

```
Admin@Monica MINGW64 ~/git-assignment (feature)
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
```

Step 7: Add and commit the changes to the master branch.

Step 8: Push the changes to the master branch.

Note: There will be a conflict in the pull request, how do we resolve it??

Hint: git rebase

A new window opened with

modified the file for 3rd task # Conflicts: # file1.txt # Please enter the commit message for your changes. Lines starting # with '#' will be ignored, and an empty message aborts the commit. # # interactive rebase in progress; onto f22de4e # Last command done (1 command done): # pick 3be60eb modified the file for 3rd task # No commands remaining. # You are currently rebasing branch 'feature' on 'f22de4e'. # # Changes to be committed: # modified: file1.txt what is this

so there was nothing to edit so I used esc -> : ->wq

```
[detached HEAD 7843939] modified the file for 3rd task
 1 file changed, 4 insertions(+)
Successfully rebased and updated refs/heads/feature.
```

Now Because I have rebased and changed history, Git requires a force push.

```
Admin@Monica MINGW64 ~/git-assignment (feature)
$ git push origin feature --force
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 392 bytes | 392.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/aredwin/git-assignment.git
 + 3be60eb...7843939 feature -> feature (forced update)
```

Question 4

Step 1: Step 1: Create a feature branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature)
$ git branch feature-cherry
```

Step 2: Switch to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature)
$ git checkout feature-cherry
Switched to branch 'feature-cherry'
```

open the file and make some changes to it.

```
Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ echo "this is the task-4" > file1.txt
```

Add and commit the changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the nex
t time Git touches it

Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ git commit -m "this is for feature-cherry branch"
[feature-cherry f1a1835] this is for feature-cherry branch
 1 file changed, 1 insertion(+), 5 deletions(-)
```

open the same file and make some changes to it.

```
Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ echo "this is for commit2" >>file1.txt
```

Add and commit the changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the nex
t time Git touches it

Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ git commit -m "this is for commit2 in feature-cherry branch"
[feature-cherry 4364701] this is for commit2 in feature-cherry branch
 1 file changed, 1 insertion(+)
```

open the same file and make some changes to it.

```
Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ echo "this is for commit3" >>file1.txt
```

Add and commit the changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature-cherry)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the nex
t time Git touches it
```

```
[master 318d920] this is for commit2 in feature-cherry branch
 Date: Tue May 20 22:59:31 2025 +0530
 1 file changed, 5 insertions(+)
```

Step 3: Identify the commit or commits that you want to "cherry-pick"(Note the hash of the commit or commits that you want to "cherry-pick")

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git log --oneline
318d920 (HEAD -> master, origin/master) this is for commit2 in feature-cherry branch
f22de4e content is added from master branch
7015bd5 file has been merged
60c1378 (local) added the content in the new branch
19a3a4e (origin/main, origin/HEAD, main) Initial commit
31e9c2a modified the changes
709ad83 this is v1
```

Step 4: Use the "git checkout" commagitnd to switch to the branch where you want to apply the changes.

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git cherry-pick 4364701
```

Step 5: Use the "git cherry-pick" command followed by the commit hash(es) that you want to apply.

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 412 bytes | 206.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/aredwin/git-assignment.git
   f22de4e..318d920  master -> master
```

Question 5

Step 1: Step 1: Create a feature branch.

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git branch feature2
```

Step 2: Switch to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (master)
$ git checkout feature2
Switched to branch 'feature2'
```

open the file and make some changes to it.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ echo "first commit" >> file1.txt
```

Add and commit the changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git commit -m "this is for commit-1"
[feature2 c4f333d] this is for commit-1
 1 file changed, 1 insertion(+)
```

open the same file and make some changes to it.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ echo "second commit" >> file1.txt
```

Add and commit the changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git commit -m "this is for commit-2"
[feature2 302490b] this is for commit-2
 1 file changed, 1 insertion(+)
```

open the same file and make some changes to it.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ echo "third commit" >> file1.txt
```

Add and commit the changes to the new branch.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git add file1.txt
warning: in the working copy of 'file1.txt', LF will be replaced by CRLF the next time Git touches it

Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git commit -m "this is for commit-3"
[feature2 31bc267] this is for commit-3
 1 file changed, 1 insertion(+)
```

Step 3: Use the "git log" command to view the commit history and identify the commit to which you want to reset.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git log --oneline
31bc267 (HEAD -> feature2) this is for commit-3
302490b this is for commit-2
c4f333d this is for commit-1
318d920 (origin/master, master) this is for commit2 in feature-cherry branch
f22de4e content is added from master branch
7015bd5 file has been merged
60c1378 (local) added the content in the new branch
19a3a4e (origin/main, origin/HEAD, main) Initial commit
31e9c2a modified the changes
709ad83 this is v1
```

Step 4: Use the "git reset" command followed by the desired reset type and the commit hash

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git reset --soft c4f333d
```

Step 5: Verify that the reset was successful by using the "git log" command again.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git log --oneline
c4f333d (HEAD -> feature2) this is for commit-1
318d920 (origin/master, master) this is for commit2 in feature-cherry branch
f22de4e content is added from master branch
7015bd5 file has been merged
60c1378 (local) added the content in the new branch
19a3a4e (origin/main, origin/HEAD, main) Initial commit
31e9c2a modified the changes
709ad83 this is v1
```

Step 6: Use the "git log" command to view the commit history and identify the commit that you want to reverse.

```
Admin@Monica MINGW64 ~/git-assignment (feature2)
$ git log --oneline
b5d9db6 (HEAD -> feature2) reverted
31ad613 this is for revert the commit
c4f333d this is for commit-1
318d920 (origin/master, master) this is for commit2 in feature-cherry branch
f22de4e content is added from master branch
Revert "this is for revert the commit"
```

Step 7: Use the "git revert" command followed by the commit hash or reference to which you want to revert. (Hint: git revert <commit hash>)

```
This reverts commit 31ad613f925a7331e16ec733cafb6e3bfdd09a4d.

# Conflicts:
#       file1.txt

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch feature2
# You are currently reverting commit 31ad613.
#
# Changes to be committed:
#       modified:   file1.txt
#
```

Step 8: Verify that the revert was successful by using the "git log" command again.

```
[feature2 dcd45e5] Revert "this is for revert the commit"
 1 file changed, 3 insertions(+)
```

Note: Identify the difference between git log after git reset and git r evert.

Feature	git reset	git revert
Affects history	Yes (removes commits)	No (adds a new "undo" commit)
Safe for shared repos	no	yes
Keeps file changes	Optional (--soft, --mixed)	No (creates a new commit to undo)