WEEK 2 Scenario basedquestions ANSWER

( Note: check with question for answer, not with question number for answer)

**Scenario based questions on Basic git commands**

**1. You modified a file but want to undo the changes before staging it.**

**Question:**  
You made changes to main.py but haven’t staged them yet. You realize they were a mistake. What Git command will you use to discard the local changes?

✅ **Concept:** Discarding unstaged changes  
💬 **Expected Answer:**

bash

CopyEdit

git checkout -- main.py

*or in modern Git:*

bash

CopyEdit

git restore main.py

**🔹 2. You staged changes but want to unstage them.**

**Question:**  
You accidentally ran git add file1.txt, but you’re not ready to commit yet. How do you remove it from the staging area without losing the changes?

✅ **Concept:** Unstaging files  
💬 **Expected Answer:**

bash

CopyEdit

git reset file1.txt

*or:*

bash

CopyEdit

git restore --staged file1.txt

**🔹 3. You committed with the wrong message.**

**Question:**  
You made a commit but typed the wrong commit message. You haven’t pushed it yet. How do you fix it?

✅ **Concept:** Amending commits  
💬 **Expected Answer:**

bash

CopyEdit

git commit --amend -m "Corrected commit message"

**🔹 4. You want to see the history of changes.**

**Question:**  
You want to view the commit history of the current branch in a readable format. What Git command should you use?

✅ **Concept:** Viewing commit history  
💬 **Expected Answer:**

bash

CopyEdit

git log --oneline

*or:*

bash

CopyEdit

git log

**🔹 5. You cloned a repo but forgot to set your username/email.**

**Question:**  
After cloning a repo, how can you set your name and email globally for all Git repositories?

✅ **Concept:** Git configuration  
💬 **Expected Answer:**

bash

CopyEdit

git config --global user.name "Your Name"

git config --global user.email "your@email.com"

**🔹 6. You want to check what has changed but not staged yet.**

**Question:**  
You’ve made some edits to your files but haven’t staged them. How can you view the changes?

✅ **Concept:** Checking file differences  
💬 **Expected Answer:**

bash

CopyEdit

git diff

**🔹 7. You want to switch to another branch.**

**Question:**  
You’re on the main branch but need to switch to feature/login. What command do you use?

✅ **Concept:** Branch navigation  
💬 **Expected Answer:**

bash

CopyEdit

git checkout feature/login

*or:*

bash

CopyEdit

git switch feature/login

**🔹 8. You accidentally deleted a branch locally.**

**Question:**  
You deleted the feature-ui branch by mistake. You haven’t pushed the deletion. How can you recover it?

✅ **Concept:** Recovering deleted branches  
💬 **Expected Answer:**

bash

CopyEdit

git reflog

git checkout -b feature-ui <commit\_hash>

**🔹 9. You want to push your local commits to the remote repo.**

**Question:**  
You’ve made some commits locally and now want to upload them to the remote repository. What do you run?

✅ **Concept:** Pushing commits  
💬 **Expected Answer:**

bash

CopyEdit

git push origin <branch-name>

**🔹 10. You want to download the latest changes from the remote without merging.**

**Question:**  
How can you fetch the latest changes from the remote repository without merging them automatically?

✅ **Concept:** Fetching updates  
💬 **Expected Answer:**

bash

CopyEdit

git fetch origin

**11. You want to create a new branch and switch to it immediately.**

**Question:**  
You’re on the main branch and want to start working on a new feature called search-filter. What command do you use?

✅ **Concept:** Branch creation and switching  
💬 **Expected Answer:**

bash

CopyEdit

git checkout -b search-filter

*or in modern Git:*

bash

CopyEdit

git switch -c search-filter

**🔹 12. You accidentally committed a sensitive file. You want to remove it from Git history.**

**Question:**  
You committed a file containing an API key and want to completely remove it from the repo's history. What should you do?

✅ **Concept:** Removing sensitive data  
💬 **Expected Answer:**  
Use tools like git filter-branch or BFG Repo-Cleaner to scrub the history.  
For example:

bash

CopyEdit

git filter-branch --force --index-filter \

"git rm --cached --ignore-unmatch secrets.txt" \

--prune-empty --tag-name-filter cat -- --all

**🔹 13. You want to list all local and remote branches.**

**Question:**  
You want to see all the branches that exist both locally and on the remote. What command do you use?

✅ **Concept:** Branch listing  
💬 **Expected Answer:**

bash

CopyEdit

git branch -a

**🔹 14. You want to merge changes from another branch.**

**Question:**  
You’re on main and want to merge the completed feature/signup branch into it. What command do you use?

✅ **Concept:** Merging branches  
💬 **Expected Answer:**

bash

CopyEdit

git merge feature/signup

**🔹 15. You want to resolve a merge conflict manually.**

**Question:**  
You tried to merge two branches and Git reported a conflict in app.js. What are the general steps to resolve it?

✅ **Concept:** Conflict resolution  
💬 **Expected Answer:**

1. Open app.js and resolve the conflict markers (<<<<<<<, =======, >>>>>>>)
2. After resolving:

bash

CopyEdit

git add app.js

git commit # If Git didn’t auto-create a merge commit

**🔹 16. You want to ignore some files from tracking.**

**Question:**  
You don’t want Git to track changes to .log files or node\_modules/. How do you achieve this?

✅ **Concept:** .gitignore  
💬 **Expected Answer:**  
Create or edit a .gitignore file:

bash

CopyEdit

\*.log

node\_modules/

**🔹 17. You want to see who changed a particular line in a file.**

**Question:**  
You're investigating a bug and want to know who last changed line 25 in script.py. What command do you use?

✅ **Concept:** Blame  
💬 **Expected Answer:**

bash

CopyEdit

git blame script.py

**🔹 18. You want to stash your changes temporarily.**

**Question:**  
You’re in the middle of working on a file but need to switch branches quickly. What do you do to save your work?

✅ **Concept:** Stashing  
💬 **Expected Answer:**

bash

CopyEdit

git stash

git checkout other-branch

**🔹 19. You want to apply your last stash.**

**Question:**  
You previously ran git stash and now want to restore those changes. What command do you use?

✅ **Concept:** Applying stashed changes  
💬 **Expected Answer:**

bash

CopyEdit

git stash apply

**🔹 20. You want to delete a local branch.**

**Question:**  
The feature/test branch is no longer needed. How do you delete it locally?

✅ **Concept:** Deleting branches  
💬 **Expected Answer:**

bash

CopyEdit

git branch -d feature/test

*Use -D to force-delete if it hasn't been merged.*

## ✅ Quick Summary

| **Task** | **Command** |
| --- | --- |
| Delete local branch (safe) | git branch -d feature-ui |
| Delete local branch (force) | git branch -D feature-ui |
| Delete remote branch | git push origin --delete feature-ui |

## Scenario 1: Delete a fully merged local branch

**Question:**  
You just merged the feature-ui branch into main. You want to clean up your local branches. How do you safely delete feature-ui?

✅ **Expected Answer:**

bash

CopyEdit

git branch -d feature-ui

**Explanation:**  
-d safely deletes a branch **only if it has been fully merged** into your current branch.

## 🔹 Scenario 2: Delete a branch that hasn't been merged

**Question:**  
You created a branch feature-experiment, made some changes, but now realize the code is no longer needed. You want to delete it, even though it hasn’t been merged. What command will you use?

✅ **Expected Answer:**

bash

CopyEdit

git branch -D feature-experiment

**Explanation:**  
-D is a shortcut for --delete --force, used when you want to remove a branch regardless of merge status.

## 🔹 Scenario 3: You're not on the branch you want to delete

**Question:**  
You're currently on the feature-login branch and want to delete feature-ui. What must you ensure before running the delete command?

✅ **Expected Answer:**  
You **must not be on the branch you want to delete**. Switch to another branch first:

bash

CopyEdit

git checkout main

git branch -d feature-ui

**Explanation:**  
Git won’t let you delete the branch you’re currently on.

## 🔹 Scenario 4: You don’t remember if the branch was merged

**Question:**  
You want to delete bugfix-footer, but you're unsure if it's been merged. What should you do before deleting it?

✅ **Expected Answer:**  
Check its merge status:

bash

CopyEdit

git branch --merged

If it's listed, you can safely delete it with:

bash

CopyEdit

git branch -d bugfix-footer

## 🔹 Scenario 5: Delete multiple local branches at once

**Question:**  
You finished working on feature-a, feature-b, and feature-c. All have been merged into main. How do you delete them in one command?

✅ **Expected Answer:**

bash

CopyEdit

git branch -d feature-a feature-b feature-c

Working with Git remote commands

### 1. ****Cloning a Remote Repository****

When you're starting work on a project and need to clone a remote repository to your local machine:

bash

CopyEdit

git clone <repository\_url>

Example:

bash

CopyEdit

git clone https://github.com/username/repository.git

### 2. ****Checking Existing Remotes****

If you want to see the remotes that are connected to your local repository:

bash

CopyEdit

git remote -v

This shows the URL of the remote repositories linked to your local repository.

### 3. ****Adding a Remote Repository****

If you want to add a new remote repository to your local repository:

bash

CopyEdit

git remote add <remote\_name> <repository\_url>

Example:

bash

CopyEdit

git remote add origin https://github.com/username/repository.git

### 4. ****Removing a Remote****

To remove a remote repository from your local configuration:

bash

CopyEdit

git remote remove <remote\_name>

Example:

bash

CopyEdit

git remote remove origin

### 5. ****Renaming a Remote****

If you want to rename an existing remote:

bash

CopyEdit

git remote rename <old\_name> <new\_name>

Example:

bash

CopyEdit

git remote rename origin upstream

### 6. ****Fetching Updates from Remote****

To fetch updates from the remote repository but not merge them into your local branch:

bash

CopyEdit

git fetch <remote\_name>

Example:

bash

CopyEdit

git fetch origin

### 7. ****Pulling Updates from Remote****

To pull the latest changes from the remote repository and merge them into your local branch:

bash

CopyEdit

git pull <remote\_name> <branch\_name>

Example:

bash

CopyEdit

git pull origin main

### 8. ****Pushing Changes to Remote****

To push your local commits to a remote repository:

bash

CopyEdit

git push <remote\_name> <branch\_name>

Example:

bash

CopyEdit

git push origin main

### 9. ****Setting the Upstream Branch for Pushing****

When pushing for the first time and want to set the remote branch you are pushing to:

bash

CopyEdit

git push --set-upstream <remote\_name> <branch\_name>

Example:

bash

CopyEdit

git push --set-upstream origin feature-branch

### 10. ****Changing Remote URL****

To change the URL of a remote (e.g., after changing the remote repository address):

bash

CopyEdit

git remote set-url <remote\_name> <new\_url>

Example:

bash

CopyEdit

git remote set-url origin https://github.com/username/new-repository.git

### 11. ****Listing All Remote Branches****

If you want to list all branches on the remote repository:

bash

CopyEdit

git branch -r

### 12. ****Pruning Deleted Remotes****

If a branch was deleted on the remote but still shows up locally:

bash

CopyEdit

git remote prune <remote\_name>

Example:

bash

CopyEdit

git remote prune origin

### 13. ****Fetching a Specific Remote Branch****

To fetch a specific branch from a remote:

bash

CopyEdit

git fetch <remote\_name> <branch\_name>

Example:

bash

CopyEdit

git fetch origin feature-branch

### 14. ****Viewing the Remote Repository’s Information****

If you want to see detailed information about a remote repository:

bash

CopyEdit

git remote show <remote\_name>

Example:

bash

CopyEdit

git remote show origin

### 15. ****Rebasing a Local Branch onto a Remote Branch****

If you want to rebase your local branch onto a remote branch (this can be useful to keep your history linear):

bash

CopyEdit

git fetch <remote\_name>

git rebase <remote\_name>/<branch\_name>

Example:

bash

CopyEdit

git fetch origin

git rebase origin/main