■ GIT BRANCHING — STUDENT WORKSHEET

■ Learn how to create, switch, merge, and manage branches confidently.

■ Part 1 — Start Fresh

Goal: Create a new Git project and make your first commit.

Commands:

```
mkdir git-branching-practice
cd git-branching-practice
git init
echo "Hello Git branching!" > notes.txt
git add notes.txt
git commit -m "Initial commit"
git log --oneline
```

■ Task 1: Write down your first commit ID and describe what ‘git init’ does.

■ Part 2 — Create and Switch Branches

Goal: Make a new branch for a new feature.

```
git checkout -b feature-1
echo "Working on feature 1" >> notes.txt
git add notes.txt
git commit -m "Add feature 1"
```

■ Task 2: Which branch are you currently on? Write the command to check.

■ Part 3 — Merge a Branch

Goal: Merge your feature back into main.

```
git checkout main
git merge feature-1
```

■ Task 3: What type of merge did Git perform (fast-forward or recursive)?

■■ Part 4 — Simulate a Merge Conflict

Goal: Learn to handle conflicts manually.

```
git checkout -b feature-2
echo "Feature 2 edit" >> notes.txt
git commit -am "Feature-2 edit"
git checkout main
echo "Main edit" >> notes.txt
git commit -am "Main edit"
git merge feature-2
```

■ Task 4: Open the conflicted file. Write the conflict markers you see and explain how you resolved it.

■ Part 5 — Clean Up

Goal: Delete merged branches.

git branch -d feature-1
git branch -d feature-2

■ Task 5: After deletion, which branches remain? Use ‘git branch’ to confirm.

■ Review Questions

1. What’s the purpose of creating branches?

2. What’s the difference between ‘git branch’ and ‘git checkout -b’?

3. When does a fast-forward merge happen?

4. What is a merge conflict and how do you resolve it?

5. Why might you delete a branch after merging?

■ Final Challenge

Create this branching structure and merge in the correct order:

main → feature-A → feature-B → bugfix → main

■ Rules:

• Each branch adds one line to notes.txt.
• Commit after each edit.
• Merge step by step.
• Resolve any conflicts cleanly.

At the end, run ‘git log --oneline --graph --all’ and draw your final branch diagram below:

■ Key Commands Summary

Command	Purpose
git branch name	Create a new branch
git checkout name	Switch branches
git checkout -b name	Create and switch in one step
git merge name	Merge a branch
git branch -d name	Delete a branch
git log --oneline --graph --all	Visualize branch structure