Submiited by -**Rithika**

Need for Merging

- Multiple developers work on different branches and later combine work.
- To combine a topic-branch (feature/fix) into the integration branch (main/develop).
- To integrate bugfixes across release lines.
- To keep branches synchronized so integration problems are discovered early.

Merge Strategies in Git

1. Three-Way Merge:

- They have 3 version – base, current, incoming
- Initially file name – "Hello World" commit
- In Feature branch –"Hello from feature a" commit
- Again in main – "Hello Git world " commit
- On using : **git merge feature , conflict** occurs
- If both branches changed the SAME line differently.

2. Recursive Merge Strategy – Merge a single source branch into current branch

- Git finds the **merge base** (common ancestor commit) and performs a three-way merge between: merge-base, current branch tip, and other

Initialize repository -> Configure name/email -> Create initial commit on main-> Create feature branch and commit -> Switch back to main and add a new commit -> Merge feature branch into main with --no-ff = **Recursive merge conflict"**

Commands:

mkdir git-recursive-demo

cd git-recursive-demo

git init

git config user.name "Rithika"

git config user.email "email"

echo "Line 1 from main" > file.txt

git add file.txt

git commit -m "Initial commit on main"

git checkout -b feature-a

echo "Feature A line 1" >> file.txt

git add file.txt

git commit -m "Feature-a: add first line"

git checkout main

echo "Main line 2" >> file.txt

git add file.txt

git commit -m "Main: add second line"

git merge --no-ff feature-a -m "Merge feature-a into main using recursive merge"

git status

Resolve the conflict

git add file.txt

git log --graph --oneline --all

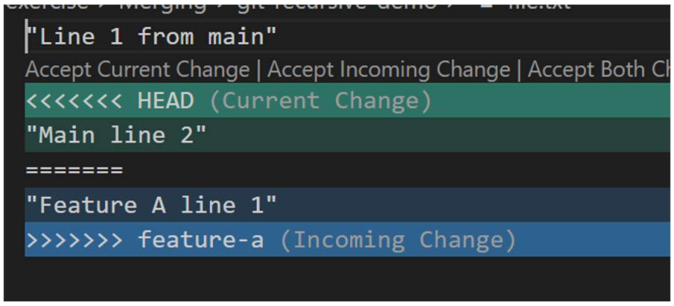
git commit -m "Merge feature-a into main, resolve conflict"

```
D:\EI exercise\Merging>mkdir git-recursive-demo

D:\EI exercise\Merging>cd git-recursive-demo

D:\EI exercise\Merging\git-recursive-demo>git init
Initialized empty Git repository in D:/EI exercise/Merging/git-recursive-dem
o/.git/

D:\EI exercise\Merging\git-recursive-demo>git config user.name "Rithika"

D:\EI exercise\Merging\git-recursive-demo>git config user.email "rithikasent
hilkumar4@gmail.com"

D:\EI exercise\Merging\git-recursive-demo>echo "Line 1 from main" > file.txt


D:\EI exercise\Merging\git-recursive-demo>git add file.txt

D:\EI exercise\Merging\git-recursive-demo>git commit -m "Initial commit on m
ain"
[master (root-commit) b988090] Initial commit on main
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
```

```
D:\EI exercise\Merging\git-recursive-demo>git checkout -b feature-a
Switched to a new branch 'feature-a'

D:\EI exercise\Merging\git-recursive-demo>echo "Feature A line 1" >> file.tx
t

D:\EI exercise\Merging\git-recursive-demo>git add file.txt
```

```
D:\EI exercise\Merging\git-recursive-demo>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\git-recursive-demo>echo "Main line 2" >> file.txt

D:\EI exercise\Merging\git-recursive-demo>git add file.txt

D:\EI exercise\Merging\git-recursive-demo>git commit -m "Main: add second li
ne"
[master d2a88a3] Main: add second line
 1 file changed, 1 insertion(+)

D:\EI exercise\Merging\git-recursive-demo>git merge --no-ff feature-a -m "Me
rge feature-a into main using recursive merge"
Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Open the conflicted file -> Conflicted markers -> remove them and save



the merge commit has two parents

```
D:\EI exercise\Merging\git-recursive-demo>git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   file.txt

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

D:\EI exercise\Merging\git-recursive-demo>git add file.txt

D:\EI exercise\Merging\git-recursive-demo>git commit -m "Merge feature-a i
o main, resolve conflict"
[master 1ff24bc] Merge feature-a into main, resolve conflict

D:\EI exercise\Merging\git-recursive-demo>git log --graph --oneline --all
*   1ff24bc (HEAD -> master) Merge feature-a into main, resolve conflict
|\
| * 504cdae (feature-a) Feature-a: add first line
* | d2a88a3 Main: add second line
|/
* b988090 Initial commit on main
```

Note:

THREE-WAY MERGE = method used to combine contents

RECURSIVE MERGE = strategy used to merge branches using three-way merge repeatedly

3.) Fast Forward Merge: (Linear) - Moves main pointer to branch head

- When the target branch has not diverged from the source branch. In this case, they simply moves the target branch pointer to the latest commit in the source branch - main had **no new commits** after branching

Initialize Repo -> Create first commit on main -> Create a new branch -> Make some changes in feature branch -> Now merge back into main

Git just moved the pointer forward

Before merge:

main -> a1

feature-a -> b1

After merge:

main -> b1

feature-a -> b1

Commands:

mkdir git-ff-demo

cd git-ff-demo

git init

echo "Initial Line" > file.txt

git add .

git commit -m "Initial commit"

git checkout -b feature-a

echo "Added feature A" >> file.txt

git add file.txt

git commit -m "Add feature A work"

git log --oneline --graph –all

git checkout main

git log –oneline

git merge feature-a

```
D:\EI exercise\Merging>mkdir git-ff-demo

D:\EI exercise\Merging>cd git-ff-demo

D:\EI exercise\Merging\git-ff-demo>git init
Initialized empty Git repository in D:/EI exercise/Merging/git-ff-demo/.git/

D:\EI exercise\Merging\git-ff-demo>echo "Initial Line" > file.txt

D:\EI exercise\Merging\git-ff-demo>git add .

D:\EI exercise\Merging\git-ff-demo>git commit -m "Initial commit"
[master (root-commit) 54b56f2] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt
```

```
D:\EI exercise\Merging\git-ff-demo>git checkout -b feature-a
Switched to a new branch 'feature-a'

D:\EI exercise\Merging\git-ff-demo>echo "Added feature A" >> file.txt

D:\EI exercise\Merging\git-ff-demo>git add file.txt

D:\EI exercise\Merging\git-ff-demo>git commit -m "Add feature A work"
[feature-a 0e162e2] Add feature A work
 1 file changed, 1 insertion(+)

D:\EI exercise\Merging\git-ff-demo>git log --oneline --graph --all
* 0e162e2 (HEAD -> feature-a) Add feature A work
* 54b56f2 (master) Initial commit
```

```
D:\EI exercise\Merging\git-ff-demo>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\git-ff-demo>git log --oneline
54b56f2 (HEAD -> master) Initial commit

D:\EI exercise\Merging\git-ff-demo>git merge feature-a
Updating 54b56f2..0e162e2
Fast-forward
 file.txt | 1 +
 1 file changed, 1 insertion(+)

D:\EI exercise\Merging\git-ff-demo>git log --oneline --graph --all
* 0e162e2 (HEAD -> master, feature-a) Add feature A work
* 54b56f2 Initial commit
```

4.) Squash and Merge:

- Compress as 1 commit
- Collects all changes from the feature branch and stages them in your working tree as a **single** set of changes

1) Create repo and initial commit

mkdir squash-demo

cd squash-demo

git init

echo "Line 1" > file.txt

git add file.txt

git commit -m "Initial commit"

2) Create feature branch and make multiple small commits

git checkout -b feature-squash

echo "Feature line A" >> file.txt

```
git add file.txt
git commit -m "feat: add A"
echo "Feature line B" >> file.txt
git add file.txt
git commit -m "feat: add B"
echo "Feature line C" >> file.txt
git add file.txt
git commit -m "chore: add C"
git log --oneline --graph --decorate –all
```

```
D:\EI exercise\Merging>mkdir squash-demo

D:\EI exercise\Merging>cd squash-demo

D:\EI exercise\Merging\squash-demo>git init
Initialized empty Git repository in D:/EI exercise/Merging/squash-demo/.git/

D:\EI exercise\Merging\squash-demo>echo "Line 1" > file.txt

D:\EI exercise\Merging\squash-demo>git add file.txt

D:\EI exercise\Merging\squash-demo>git commit -m "Initial commit"
[master (root-commit) e61b7fa] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

D:\EI exercise\Merging\squash-demo>git checkout -b feature-squash
Switched to a new branch 'feature-squash'

D:\EI exercise\Merging\squash-demo>echo "Feature line A" >> file.txt

D:\EI exercise\Merging\squash-demo>git add file.txt

D:\EI exercise\Merging\squash-demo>git commit -m "feat: add A"
[feature-squash 50789f7] feat: add A
 1 file changed, 1 insertion(+)

D:\EI exercise\Merging\squash-demo>echo "Feature line B" >> file.txt

D:\EI exercise\Merging\squash-demo>git add file.txt

D:\EI exercise\Merging\squash-demo>git commit -m "feat: add B"
[feature-squash d5a1ca2] feat: add B
 1 file changed, 1 insertion(+)
```

```
D:\EI exercise\Merging\squash-demo>git log --oneline --graph --decorate --al
l
* d5a1ca2 (HEAD -> feature-squash) feat: add B
* 50789f7 feat: add A
* e61b7fa (master) Initial commit

D:\EI exercise\Merging\squash-demo>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\squash-demo>git merge --squash feature-squash
Updating e61b7fa..d5a1ca2
Fast-forward
Squash commit -- not updating HEAD
 file.txt | 2 ++
 1 file changed, 2 insertions(+)

D:\EI exercise\Merging\squash-demo>git commit -m "feat: add feature-squash (
squashed)"
[master fdec47a] feat: add feature-squash (squashed)
 1 file changed, 2 insertions(+)

D:\EI exercise\Merging\squash-demo>git log --oneline --graph --decorate --al
l
* fdec47a (HEAD -> master) feat: add feature-squash (squashed)
| * d5a1ca2 (feature-squash) feat: add B
| * 50789f7 feat: add A
|/
* e61b7fa Initial commit
```

Scenario based Ex: Developing a Search Feature

All 2 development commits(add btn, debug) are now collapsed into 1 clean commit

```
D:\EI exercise\Merging>mkdir squash-example

D:\EI exercise\Merging>cd squash-example

D:\EI exercise\Merging\squash-example>git init
Initialized empty Git repository in D:/EI exercise/Merging/squash-example/.g
it/

D:\EI exercise\Merging\squash-example>echo "Home Page" > index.html

D:\EI exercise\Merging\squash-example>git add .

D:\EI exercise\Merging\squash-example>git commit -m "Initial project structu
re"
[master (root-commit) dcb27db] Initial project structure
 1 file changed, 1 insertion(+)
 create mode 100644 index.html

D:\EI exercise\Merging\squash-example>git checkout -b feature/search
Switched to a new branch 'feature/search'

D:\EI exercise\Merging\squash-example>
D:\EI exercise\Merging\squash-example>echo "<input type='text' id='search' /
>" >> index.html

D:\EI exercise\Merging\squash-example>git add index.html

D:\EI exercise\Merging\squash-example>git commit -m "feat: add search UI"
[feature/search eeac66a] feat: add search UI
 1 file changed, 1 insertion(+)
```

```
D:\EI exercise\Merging\squash-example>echo "<script>console.log('debug');</s
cript>" >> index.html

D:\EI exercise\Merging\squash-example>git add index.html

D:\EI exercise\Merging\squash-example>git commit -m "debug: add log for debu
gging"
[feature/search 41eb5fa] debug: add log for debugging
 1 file changed, 1 insertion(+)
```

```
D:\EI exercise\Merging\squash-example>git add index.html

D:\EI exercise\Merging\squash-example>git commit -m "refactor: remove debug
logs"
On branch feature/search
nothing to commit, working tree clean

D:\EI exercise\Merging\squash-example>git log --oneline
41eb5fa (HEAD -> feature/search) debug: add log for debugging
eeac66a feat: add search UI
dcb27db (master) Initial project structure

D:\EI exercise\Merging\squash-example>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\squash-example>git merge --squash feature/search
Updating dcb27db..41eb5fa
Fast-forward
Squash commit -- not updating HEAD
 index.html | 2 ++
 1 file changed, 2 insertions(+)
```

```
D:\EI exercise\Merging\squash-example>git commit -m "feat: Add Search Featur
e"
[master a1a92d8] feat: Add Search Feature
 1 file changed, 2 insertions(+)

D:\EI exercise\Merging\squash-example>git log --oneline --graph --all
* a1a92d8 (HEAD -> master) feat: Add Search Feature
| * 41eb5fa (feature/search) debug: add log for debugging
| * eeac66a feat: add search UI
|/
* dcb27db Initial project structure

D:\EI exercise\Merging\squash-example>git show a1a92d8
commit a1a92d8cf89a44e46798e0cd8e0a1f4b64465647 (HEAD -> master)
Author: rithikas20 <rithikasenthilkumar4@gmail.com>
Date:   Sat Dec 6 16:52:28 2025 +0530

    feat: Add Search Feature

diff --git a/index.html b/index.html
index 0173875..f195b43 100644
--- a/index.html
+++ b/index.html
@@ -1 +1,3 @@
 "Home Page"
+"<input type='text' id='search' />"
+"<script>console.log('debug');</script>"
```

5.) Octopus Merge:

- Merging more than two branches simultaneously.
- Ex: There are three small feature branches , All features are **independent** and **do not touch the same files**, so they won't cause conflicts
- Instead of merging them **one-by-one**, you want to merge them **together** into main using **octopus merge**.

Commands:

mkdir octopus-merge-example

cd octopus-merge-example

git init

echo "Home Page" > index.html

git add index.html

git commit -m "Initial home page"

git checkout -b feature/login-ui

echo " Login Page UI " >> index1.html

```
git add index1.html
git commit -m "Add login UI"
git checkout main
git checkout -b feature/banner
echo "<header>Banner Section</header>" >> index2.html
git add index2.html
git commit -m "Add banner section"
git checkout main
git merge -s octopus feature/login-ui feature/banner
git log --oneline --graph –all
```

feature/login-ui

\

feature/banner -- merge --> main

/

feature/footer

Suitable When	**Not Suitable When**
Many small independent branches	Conflicts expected
Want clean merge	Manual resolution needed
Automated integrations	Shared file changes
CI/CD pipelines	Complex feature integrations

```
D:\EI exercise\Merging>mkdir octopus-merge-example

D:\EI exercise\Merging>cd octopus-merge-example

D:\EI exercise\Merging\octopus-merge-example>git init
Initialized empty Git repository in D:/EI exercise/Merging/octopus-merge-exa
mple/.git/

D:\EI exercise\Merging\octopus-merge-example>echo "Home Page" > index.html

D:\EI exercise\Merging\octopus-merge-example>git add index.html

D:\EI exercise\Merging\octopus-merge-example>git commit -m "Initial home pag
e"
[master (root-commit) 6886fab] Initial home page
 1 file changed, 1 insertion(+)
 create mode 100644 index.html

D:\EI exercise\Merging\octopus-merge-example>git checkout -b feature/login-u
i
Switched to a new branch 'feature/login-ui'

D:\EI exercise\Merging\octopus-merge-example>echo "<div>Login Page UI</div>"
 >> index1.html

D:\EI exercise\Merging\octopus-merge-example>git add index1.html

D:\EI exercise\Merging\octopus-merge-example>git commit -m "Add login UI"
[feature/login-ui a59b39d] Add login UI
 1 file changed, 1 insertion(+)
 create mode 100644 index1.html
```

```
D:\EI exercise\Merging\octopus-merge-example>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\octopus-merge-example>git checkout -b feature/banner
Switched to a new branch 'feature/banner'

D:\EI exercise\Merging\octopus-merge-example>echo "<header>Banner Section</h
eader>" >> index2.html

D:\EI exercise\Merging\octopus-merge-example>git add index2.html

D:\EI exercise\Merging\octopus-merge-example>git commit -m "Add banner secti
on"
[feature/banner 83a0098] Add banner section
 1 file changed, 1 insertion(+)
 create mode 100644 index2.html

D:\EI exercise\Merging\octopus-merge-example>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\octopus-merge-example>git merge -s octopus feature/lo
gin-ui feature/banner
Fast-forwarding to: feature/login-ui
Trying simple merge with feature/banner
Merge made by the 'octopus' strategy.
 index1.html | 1 +
 index2.html | 1 +
 2 files changed, 2 insertions(+)
 create mode 100644 index1.html
 create mode 100644 index2.html
```

```
Merge branches 'feature/login-ui' and 'feature/banner'
# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
```

6. **Subtree Merge:**

- **Pull another repository (or project)** and merge it into a **subdirectory** of your current project
- **Ex :**

Repo1

my-app/
index.html
styles.css

Repo2:

ui-library/
button.js
input.js

To merge as : my-app/libs/ui-library/

FINAL O/P :

subtree-merge-main/
app.txt
libs/
ui-library/
button.js

Commands:

```
mkdir subtree-merge-main
cd subtree-merge-main
git init
echo "Main Application" > app.txt
git add .
git commit -m "Initial commit in main project"
cd ..
mkdir subtree-ui-library
cd subtree-ui-library
git init
echo "Button Component" > button.js
git add .
git commit -m "Initial library commit"
cd ../subtree-merge-main
git remote add ui_lib ../subtree-ui-library
```

git fetch ui_lib

git merge -s subtree --allow-unrelated-histories ui_lib/master -m "Merge UI Library into libs/ui-library folder"

```
D:\EI exercise\Merging>mkdir subtree-merge-main

D:\EI exercise\Merging>cd subtree-merge-main

D:\EI exercise\Merging\subtree-merge-main>git init
Initialized empty Git repository in D:/EI exercise/Merging/subtree-merge-mai
n/.git/

D:\EI exercise\Merging\subtree-merge-main>echo "Main Application" > app.txt

D:\EI exercise\Merging\subtree-merge-main>git add .

D:\EI exercise\Merging\subtree-merge-main>git commit -m "Initial commit in m
ain project"
[master (root-commit) 138ebf3] Initial commit in main project
 1 file changed, 1 insertion(+)
 create mode 100644 app.txt
```

```
D:\EI exercise\Merging\subtree-merge-main>cd ..

D:\EI exercise\Merging>mkdir subtree-ui-library

D:\EI exercise\Merging>cd subtree-ui-library

D:\EI exercise\Merging\subtree-ui-library>git init
Initialized empty Git repository in D:/EI exercise/Merging/subtree-ui-librar
y/.git/

D:\EI exercise\Merging\subtree-ui-library>echo "Button Component" > button.j
s

D:\EI exercise\Merging\subtree-ui-library>git add .

D:\EI exercise\Merging\subtree-ui-library>git commit -m "Initial library com
mit"
[master (root-commit) a6b0fed] Initial library commit
 1 file changed, 1 insertion(+)
 create mode 100644 button.js
```

```
D:\EI exercise\Merging\subtree-ui-library>cd ../subtree-merge-main

D:\EI exercise\Merging\subtree-merge-main>git remote add ui_lib ../subtree-u
i-library

D:\EI exercise\Merging\subtree-merge-main>git fetch ui_lib
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 225 bytes | 20.00 KiB/s, done.
From ../subtree-ui-library
 * [new branch]      master     -> ui_lib/master
```

```
D:\EI exercise\Merging\subtree-merge-main>git merge -s subtree --allow-unrel
ated-histories ui_lib/master -m "Merge UI Library into libs/ui-library folde
r"
Merge made by the 'subtree' strategy.
 button.js | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 button.js

D:\EI exercise\Merging\subtree-merge-main>git log --oneline --graph --all
*   e35a020 (HEAD -> master) Merge UI Library into libs/ui-library folder
|\
| * a6b0fed (ui_lib/master, ui_lib/HEAD) Initial library commit
* 138ebf3 Initial commit in main project
```

7.) Ours strategies:

- Discards all changes from the branch being merged, but still creates a merge commit so Git knows the branches are “merged.
- Keeps all changes from the current branch

Setup main branch-> Create feature branch with conflicting changes -> Go back to main branch-> Merge using ours strategy -> Check file content -> Created a merge commit, but kept only the main branch’s content.

Commands:

mkdir ours-strategy-demo

cd ours-strategy-demo

git init

echo "Line from main" > file.txt

git add .

git commit -m "Initial commit on main"

git checkout -b feature/experimental

echo "Line from feature" >> file.txt

git add .

git commit -m "Add experimental line"

git checkout main/master

git merge -s ours feature/experimental -m "Merge feature/experimental using ours"

type file.txt

```
D:\EI exercise\Merging>mkdir ours-strategy-demo

D:\EI exercise\Merging>cd ours-strategy-demo

D:\EI exercise\Merging\ours-strategy-demo>git init
Initialized empty Git repository in D:/EI exercise/Merging/ours-strategy-dem
o/.git/

D:\EI exercise\Merging\ours-strategy-demo>echo "Line from main" > file.txt

D:\EI exercise\Merging\ours-strategy-demo>git add .

D:\EI exercise\Merging\ours-strategy-demo>git commit -m "Initial commit on m
ain"
[master (root-commit) 31e7d4b] Initial commit on main
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

D:\EI exercise\Merging\ours-strategy-demo>git checkout -b feature/experiment
al
Switched to a new branch 'feature/experimental'

D:\EI exercise\Merging\ours-strategy-demo>echo "Line from feature" >> file.t
xt

D:\EI exercise\Merging\ours-strategy-demo>git add .
```

```
D:\EI exercise\Merging\ours-strategy-demo>git commit -m "Add experimental li
ne"
[feature/experimental 1871426] Add experimental line
 1 file changed, 1 insertion(+)

D:\EI exercise\Merging\ours-strategy-demo>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\ours-strategy-demo>git merge -s ours feature/experime
ntal -m "Merge feature/experimental using ours"
Merge made by the 'ours' strategy.
```

```
D:\EI exercise\Merging\ours-strategy-demo>type file.txt
"Line from main"

D:\EI exercise\Merging\ours-strategy-demo>git log --oneline --graph --all
*   e9dd10c (HEAD -> master) Merge feature/experimental using ours
|\
| * 1871426 (feature/experimental) Add experimental line
|/
* 31e7d4b Initial commit on main
```

Merge Type	What happens to file content?	Commits in history	Real-life use case
Fast-forward	Adds feature content	Linear, no merge commit	Small, simple updates
Squash	Adds all feature content	Single commit	Keep main branch clean
Ours	Keeps main content, discards feature	Merge commit exists	Ignore unwanted feature changes

8.) **Rebase:**

- Reapplies the commit from one branch onto another ,
- No merge commit

How they Diff from merging?:

In merging:

```
A --- B -------- M  (main)
\        /
 C --- D        (feature)
```

In Rebase:

```
A --- B --- C' --- D'  (main after rebase)
```

Commands:

mkdir rebase-demo

cd rebase-demo

git init

echo "Line 1" > file.txt

git add .

git commit -m "Initial commit"

git checkout -b feature

echo "Feature line 1" >> file.txt

git add .

git commit -m "Add feature line 1"

git checkout main

echo "Main line 2" >> file.txt

git add .

git commit -m "Add main line 2"

git checkout feature

git rebase main/master

Resolve conflict manually

or use

git checkout --ours file.txt - Keep the version from the branch you're rebasing **onto**

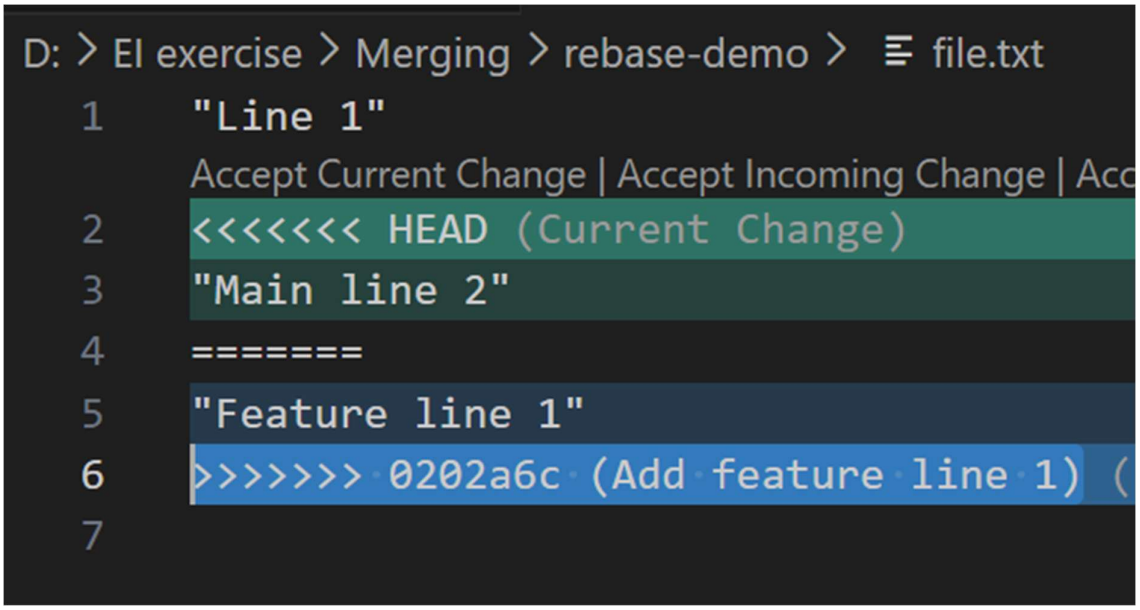
git checkout --theirs file.txt - Keep the version from the commit being applied (incoming)

git add file.txt

git rebase --continue

```
D:\EI exercise\Merging>mkdir rebase-demo

D:\EI exercise\Merging>cd rebase-demo

D:\EI exercise\Merging\rebase-demo>git init
Initialized empty Git repository in D:/EI exercise/Merging/rebase-demo/.g

D:\EI exercise\Merging\rebase-demo>echo "Line 1" > file.txt

D:\EI exercise\Merging\rebase-demo>git add .

D:\EI exercise\Merging\rebase-demo>git commit -m "Initial commit"
[master (root-commit) 633448d] Initial commit
 1 file changed, 1 insertion(+)
 create mode 100644 file.txt

D:\EI exercise\Merging\rebase-demo>git checkout -b feature
Switched to a new branch 'feature'

D:\EI exercise\Merging\rebase-demo>echo "Feature line 1" >> file.txt

D:\EI exercise\Merging\rebase-demo>git add .

D:\EI exercise\Merging\rebase-demo>git commit -m "Add feature line 1"
[feature 0202a6c] Add feature line 1
 1 file changed, 1 insertion(+)

D:\EI exercise\Merging\rebase-demo>git checkout master
Switched to branch 'master'
```

```
D:\EI exercise\Merging\rebase-demo>git checkout master
Switched to branch 'master'

D:\EI exercise\Merging\rebase-demo>echo "Main line 2" >> file.txt

D:\EI exercise\Merging\rebase-demo>git add .

D:\EI exercise\Merging\rebase-demo>git commit -m "Add main line 2"
[master 513c44f] Add main line 2
 1 file changed, 1 insertion(+)

D:\EI exercise\Merging\rebase-demo>git checkout feature
Switched to branch 'feature'

D:\EI exercise\Merging\rebase-demo>git rebase master
Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
error: could not apply 0202a6c... Add feature line 1
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
```




```
D:\EI exercise\Merging\rebase-demo>git add file.txt

D:\EI exercise\Merging\rebase-demo>git rebase --continue
[detached HEAD 372484a] Add feature line 1
 1 file changed, 4 insertions(+)
Successfully rebased and updated refs/heads/feature.

D:\EI exercise\Merging\rebase-demo>git log --oneline --graph --all
* 372484a (HEAD -> feature) Add feature line 1
* 513c44f (master) Add main line 2
* 633448d Initial commit
```

```
D:\EI exercise\Merging\rebase-demo>git rebase master
Auto-merging file.txt
CONFLICT (content): Merge conflict in file.txt
error: could not apply f417b0b... Add feature line 1
hint: Resolve all conflicts manually, mark them as resolved with
hint: "git add/rm <conflicted_files>", then run "git rebase --continue".
hint: You can instead skip this commit: run "git rebase --skip".
hint: To abort and get back to the state before "git rebase", run "git reba
--abort".
hint: Disable this message with "git config set advice.mergeConflict false"
Could not apply f417b0b... # Add feature line 1

D:\EI exercise\Merging\rebase-demo>git checkout --ours file.txt
Updated 1 path from the index

D:\EI exercise\Merging\rebase-demo>git add file.txt

D:\EI exercise\Merging\rebase-demo>git rebase --continue
Successfully rebased and updated refs/heads/feature.

D:\EI exercise\Merging\rebase-demo>git log --oneline --graph --all
* 438a812 (HEAD -> feature, master) Add main line 2
* 9a75fd8 Initial commit
```