

1)) Clona un repositorio Git con múltiples ramas.

```
Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-fast-forward-merge (main)
$ git merge add-description
Updating b60ee5d..6a6c837
Fast-forward
 README.md | 1 +
 1 file changed, 1 insertion(+)
```

Pregunta: ¿En qué situaciones recomendaría evitar el uso de `git merge --ff`? Reflexiona sobre las desventajas de este método.

No usaría cuando se tienen ramas con historiales de commits configurados anteriormente. Tampoco cuando la rama base haya recibido commits al momento de crearse la nueva rama "feature". Como desventaja no puedes preservar el contexto de fusión ni tampoco tener claridad de historial de cambios.

2))) Simula un flujo de trabajo de equipo.

```
Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-no-fast-foward-merge (main)
$ git merge --no-ff add-feature -m "Merge de add-feature con no fast-forward"
Merge made by the 'ort' strategy.
 README.md | 1 +
 1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-no-fast-foward-merge (main)
$ git log --oneline --graph
*    8142ac3 (HEAD -> main) Merge de add-feature con no fast-forward
| \
|  * 523df18 (add-feature) Implementar una nueva característica
| /
* d5127c4 Commit inicial en main
```

Pregunta: ¿Cuáles son las principales ventajas de utilizar `git merge --no-ff` en un proyecto en equipo? ¿Qué problemas podrían surgir al depender excesivamente de commits de fusión?

Haber como ventaja podria resaltar que se preserva el contexto de fusion, esto es esencial para los que desarrolladores que deseen mucha claridad en el historial de cambios (por ejemplo cuando hacemos commits o agregamos codigo de python o texto o archivo html).

3))) Crea múltiples commits en una rama.

```
Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-squash-merge (main)
$ git merge --squash add-basic-files
Updating 7f9cdb5..80cbfa5
Fast-forward
Squash commit -- not updating HEAD
  CONTRIBUTING.md | 1 +
  LICENSE.txt     | 1 +
  2 files changed, 2 insertions(+)
  create mode 100644 CONTRIBUTING.md
  create mode 100644 LICENSE.txt

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-squash-merge (main)
$ git add .

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-squash-merge (main)
$ git commit -m "Agregar documentacion estandar del repositorio"
[main 61bf9b1] Agregar documentacion estandar del repositorio
  2 files changed, 2 insertions(+)
  create mode 100644 CONTRIBUTING.md
  create mode 100644 LICENSE.txt

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-squash-merge (main)
$ git log --graph --oneline
* 61bf9b1 (HEAD -> main) Agregar documentacion estandar del repositorio
* 7f9cdb5 commit inicial en main
```

Pregunta: ¿Cuándo es recomendable utilizar una calabaza fusión? ¿Qué ventajas ofrece para proyectos grandes en Corporacion con fusiones estándar?

Es recomendable usar “git rebase-2 o “calabaza fusión” cuando deseas tener un historial de confirmaciones tipo lineal, pero también reescribir tu historial y no borrarlo como en otros casos.

4)) Resolver conflictos en una fusión sin avance rápido

```

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-squash-merge (main)
$ cd ..

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5 (main)
$ mkdir prueba-merge-conflict

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5 (main)
$ cd prueba-merge-conflict/

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git init
Initialized empty Git repository in C:/Users/Bianca/Documents/repositorio/actividad5/prueba-merge-conflict/.git/

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (master)
$ git branch -m main

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ echo "<html><body><h1>Proyecto inicial CC3S2</h1></body></html>" > index.html

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git commit -m "commit inicial del index.html en main"
[main (root-commit) cc54381] commit inicial del index.html en main
1 file changed, 1 insertion(+)
create mode 100644 index.html

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git checkout -b feature-update
Switched to a new branch 'feature-update'

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (feature-update)
$ echo "<p>.....</p>" >> index.html

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (feature-update)
$ cat index.html
<html><body><h1>Proyecto inicial CC3S2</h1></body></html>
<p>.....</p>

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (feature-update)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (feature-update)
$ git commit -m "Actualiza..."
[feature-update 412fe5d] Actualiza...
1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (feature-update)
$ git checkout main
Switched to branch 'main'

```

```

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ cat index.html
<html><body><h1>Proyecto inicial CC3S2</h1></body></html>

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ echo "<footer>Contacta aquí example@example.com</footer>" >> index.html

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git add index.html
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git commit -m"...index.html"
[main 8c8f4c7] ...index.html
1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ cat index.html
<html><body><h1>Proyecto inicial CC3S2</h1></body></html>
<footer>Contacta aquí example@example.com</footer>

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git merge --no-ff feature-update
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main|MERGING)
$ nano index.html

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main|MERGING)
$ git add index.html

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main|MERGING)
$ git commit
[main 2b7f5b1] Merge branch 'feature-update'

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git add index.html

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git commit
On branch main
nothing to commit, working tree clean

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-merge-conflict (main)
$ git log --graph --oneline
*    2b7f5b1 (HEAD -> main) Merge branch 'feature-update'
/* 412fe5d (feature-update) Actualiza...
* | 8c8f4c7 ...index.html
/*
cc54381 commit inicial del index.html en main

```

Preguntas:

- ¿Qué pasos adicionales tuviste que tomar para resolver el conflicto?
Tuve que arreglarlos manualmente, osea tecnicamente hablando abri el archivo usando el comando "nano.index.html" y cuando lo abri el archivo procedi a borrar los ">>>" , "==" y elencabezado "HEAD".
- ¿Qué estrategias podrías emplear para evitar conflictos en futuros desarrollos colaborativos?

Uhm podria revisar paso por paso los comandos que voy escribiendo en el terminal, pensando bien en lo que hago y bien concentrada , de paso que aprendo mas y no me confundo. Mas que todo seguiría el proceso usando bien la teoría aprendida, podria ver el historial y el archivo que voy agregando.

5)) Ejercicio: Comparar los historiales con git log después de diferentes fusiones

```
Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-2)
$ echo "caracteristica 2 agregada" >> version.txt

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-2)
$ git add version.txt
warning: in the working copy of 'version.txt', LF will be replaced by CRLF the next time Git touches it

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-2)
$ git commit -m"se agrega caracteristica 2"
[feature-2 47f4be4] se agrega caracteristica 2
1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-2)
$ git checkout main
Switched to branch 'main'

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git merge feature-1 --ff
Updating 07f329c..62ff1b3
Fast-forward
 version.txt | 1 +
1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git merge feature-2 --no-ff
Auto-merging version.txt
CONFLICT (content): Merge conflict in version.txt
Automatic merge failed; fix conflicts and then commit the result.

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main|MERGING)
$ git nanao.version.txt
git: 'nanao.version.txt' is not a git command. See 'git --help'.

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main|MERGING)
$ git nano version.txt
git: 'nano' is not a git command. See 'git --help'.

The most similar commands are
  annotate
  daemon

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main|MERGING)
$ nano version.txt

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main|MERGING)
$ git add version.txt

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main|MERGING)
$ git commit -m"resolver conflictos"
[main b7d26c4] resolver conflictos

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git checkout -b feature-3
Switched to a new branch 'feature-3'
```

```

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-3)
$ echo "caracteristica 3 paso 1" >> version.txt

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-3)
$ git add version.txt
warning: in the working copy of 'version.txt', LF will be replaced by CRLF the next time Git touches it

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-3)
$ git commit -m"caracteristica 3 paso 1"
[feature-3 2dfa2dd] caracteristica 3 paso 1
1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-3)
$ echo "caracteristica 3 paso 2" >> version.txt

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-3)
$ git add version.txt
warning: in the working copy of 'version.txt', LF will be replaced by CRLF the next time Git touches it

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-3)
$ git commit -m"caracteristica 3 paso 2"
[feature-3 9bbbfb9] caracteristica 3 paso 2
1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (feature-3)
$ git checkout main
Switched to branch 'main'

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git merge --squash feature-3
Updating b7d26c4..9bbbfb9
Fast-forward
Squash commit -- not updating HEAD
 version.txt | 2 ++
1 file changed, 2 insertions(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git commit -m"Agregar caracteistica 3 3n un commit"
[main fdf0e14] Agregar caracteistica 3 3n un commit
1 file changed, 2 insertions(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges --first-parent --branches
* b7d26c4 resolver conflictos

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges
* b7d26c4 resolver conflictos

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges ---decorate -all

```

```

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git commit -m"Agregar caracteistica 3 3n un commit"
[main fdf0e14] Agregar caracteistica 3 3n un commit
1 file changed, 2 insertions(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges --first-parent --branches
* b7d26c4 resolver conflictos

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges
* b7d26c4 resolver conflictos

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges ---decorate -all
error: switch `l' expects a numerical value

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges --decorate -all
error: switch `l' expects a numerical value

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges --decorate --all
* b7d26c4 resolver conflictos

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --first-parent
* fdf0e14 (HEAD -> main) Agregar caracteistica 3 3n un commit
* b7d26c4 resolver conflictos
* 62ff1b3 (feature-1) Agregar caracteistica1
* 07f329c ...

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --merges
* b7d26c4 resolver conflictos

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/prueba-compare-merge (main)
$ git log --graph --oneline --decorate --all
* fdf0e14 (HEAD -> main) Agregar caracteistica 3 3n un commit
| * 9bbbfb9 (feature-3) caracteistica 3 paso 2
| * 2dfa2dd caracteistica 3 paso 1
|/
* b7d26c4 resolver conflictos
|\
| * 47f4be4 (feature-2) se agrega caracteistica 2
* | 62ff1b3 (feature-1) Agregar caracteistica1
|/
* 07f329c ...

```

Preguntas:

- ¿Cómo se ve el historial en cada tipo de fusión?

Caso1: git log --graph --oneline --merges --first-parent -branches

Se puede apreciar el historial mas resumido, con las cosas mas importantes.

Caso 2: git log --graph --oneline -merges

Se puede apreciar únicamente un paso, que fue la resolución del conflicto. Es el historial más corto.

Caso 3: git log --graph --oneline --merges --decorate -all

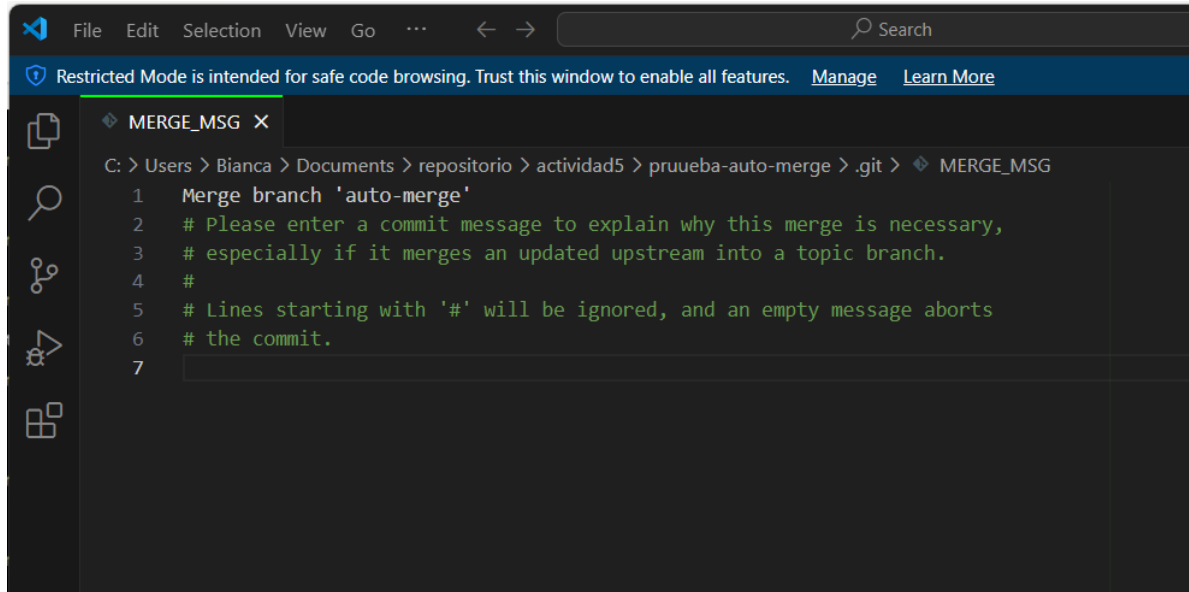
Se tiene el historial mas largo.

- ¿Qué método prefieres en diferentes escenarios y por qué?

Prefiero el método donde se usa “git log --graph --oneline --merges --first-parent --branches”, porque tiene los commits mas importantes a mi parecer.

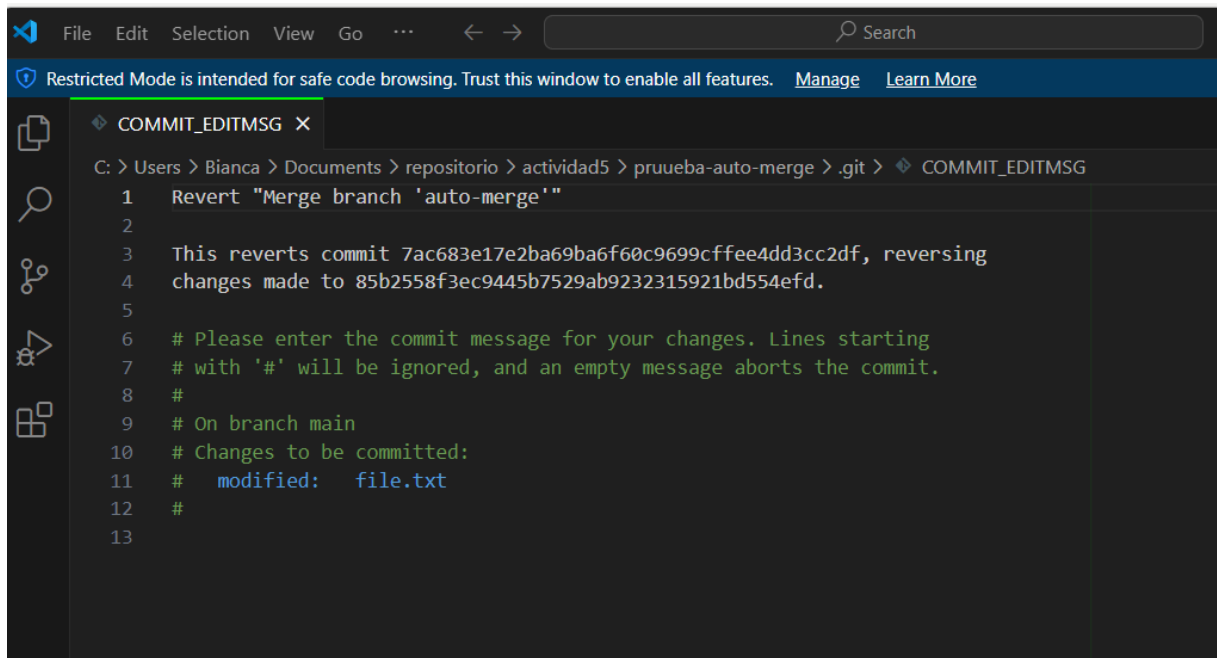
6)) Ejercicio: fusiones Usando automáticas y revertir fusiones

Acá se pudo apreciar en la interfaz de “Visual Studio Code” el archivo del “MERGE_MSG”.



The screenshot shows the Visual Studio Code interface with the 'MERGE_MSG' file open. The file path is 'C:\> Users > Bianca > Documents > repositorio > actividad5 > prueba-auto-merge > .git > MERGE_MSG'. The file content is as follows:

```
1 Merge branch 'auto-merge'
2 # Please enter a commit message to explain why this merge is necessary,
3 # especially if it merges an updated upstream into a topic branch.
4 #
5 # Lines starting with '#' will be ignored, and an empty message aborts
6 # the commit.
7
```

The image shows a Visual Studio Code editor window with a dark theme. The top menu bar includes 'File', 'Edit', 'Selection', 'View', 'Go', and a search bar. A blue banner at the top states: 'Restricted Mode is intended for safe code browsing. Trust this window to enable all features. [Manage](#) [Learn More](#)'. The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The main editor area has a tab titled 'COMMIT_EDITMSG' with a close button. The file path is 'C: > Users > Bianca > Documents > repositorio > actividad5 > prueba-auto-merge > .git > COMMIT_EDITMSG'. The content of the file is a commit message template with line numbers 1 through 13 on the left margin.

```
1 Revert "Merge branch 'auto-merge'"
2
3 This reverts commit 7ac683e17e2ba69ba6f60c9699cffee4dd3cc2df, reversing
4 changes made to 85b2558f3ec9445b7529ab9232315921bd554efd.
5
6 # Please enter the commit message for your changes. Lines starting
7 # with '#' will be ignored, and an empty message aborts the commit.
8 #
9 # On branch main
10 # Changes to be committed:
11 #   modified:   file.txt
12 #
13
```

```

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/pruueba-auto-merge (main)
$ git commit -m"agrega cabecera"
[main 85b2558] agrega cabecera
1 file changed, 1 insertion(+), 1 deletion(-)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/pruueba-auto-merge (main)
$ git merge auto-merge
Auto-merging file.txt
Merge made by the 'ort' strategy.
file.txt | 1 +
1 file changed, 1 insertion(+)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/pruueba-auto-merge (main)
$ git log --graph --oneline
* 7ac683e (HEAD -> main) Merge branch 'auto-merge'
| \
| * 4abc2e7 (auto-merge) ... linea 3
* | 85b2558 agrega cabecera
|/
* 465c068 ...linea2
* a775001 Agrega linea 1

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/pruueba-auto-merge (main)
$ cat file.txt
linea1 cabecera
linea2
linea3

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/pruueba-auto-merge (main)
$ git revert -m 1 HEAD
[main c8314e4] Revert "Merge branch 'auto-merge'"
1 file changed, 1 deletion(-)

Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/pruueba-auto-merge (main)
$ git log --graph --oneline
* c8314e4 (HEAD -> main) Revert "Merge branch 'auto-merge'"
* 7ac683e Merge branch 'auto-merge'
| \
| * 4abc2e7 (auto-merge) ... linea 3
* | 85b2558 agrega cabecera
|/
* 465c068 ...linea2
* a775001 Agrega linea 1

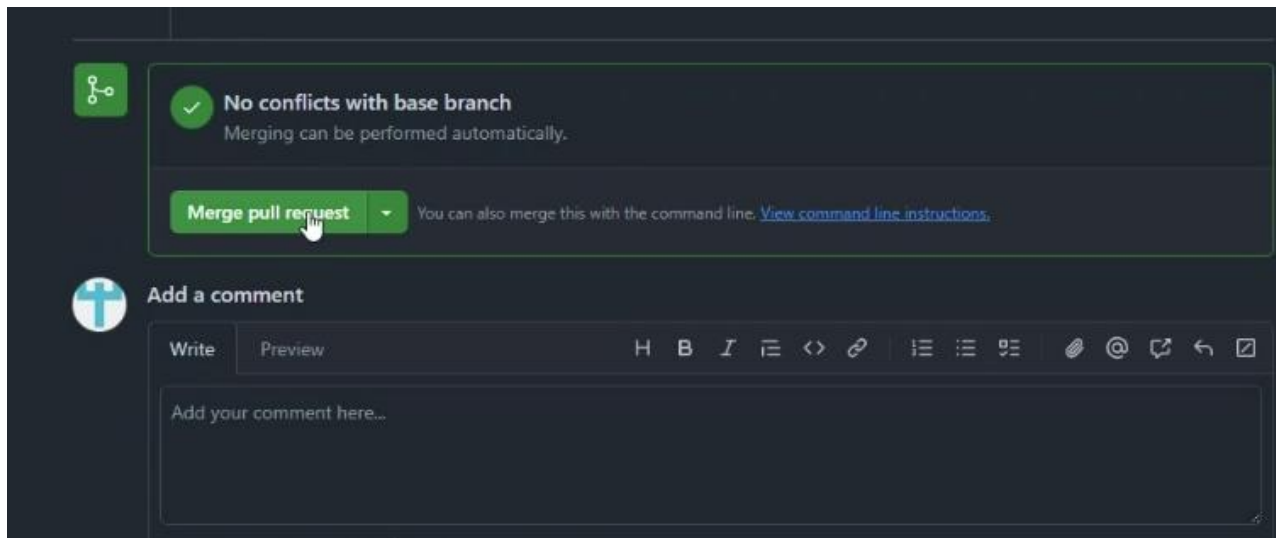
Bianca@MSI MINGW64 ~/Documents/repositorio/actividad5/pruueba-auto-merge (main)
$ cat file.txt
linea1 cabecera
linea2

```

Preguntas:

- ¿Cuándo usarías un comando como git revert para deshacer una fusión?
Cuando acab de realizar una fusion y luego me arrepiento y procedo a deshacerla sin alteral el historial.
- ¿Qué tan útil es la función de fusión automática en Git?
Demasiado util y por esa razon muchos profesionales destacados la usan.

7)))) Ejercicio: Fusión remota en un repositorio colaborativo



```
$ git push origin colaboracion
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 309 bytes | 309.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'colaboracion' on GitHub by visiting:
remote:   https://github.com/WalterRGUni/Prueba/pull/new/colaboracion
remote:
```

- ¿Cómo cambia la estrategia de fusión cuando colaboras con otras personas en un repositorio remoto?

Se deja de realizar merge directo y ahora se utiliza o se cambia por "pull request" para poder realizar la fusion de rama.

- ¿Qué problemas comunes pueden surgir al integrar ramas remotas?

Pueden ocurrir conflictos de fusion o tambien llamados "merge conflicts", tambien puedo ocasionar un historial medio incompleto o desordenado.

8)))) Ejercicio final: flujo de trabajo completo

```
$ git log --oneline --graph --all
* a962016 (HEAD -> main) Fusionando feature3 con squash
| * a09f990 (feature3) Cambios en feature3
|/
* 4907d0d Merge branch 'feature2'
| \
| * 372c62b (feature2) Cambios en feature2
* | c7f3578 (feature1) Cambios en feature1
|/
* 9a92b78 Cambios en feature1
|
```

- Revisa el historial para entender cómo los diferentes métodos de fusión afectan el árbol de commits.
- Compara los resultados y discute con tus compañeros de equipo cuál sería la mejor estrategia de fusión para proyectos más grandes.