1. ¿En qué situaciones recomendarías evitar el uso de git merge –ff ?

- Quiero preservar el historial de desarrollo de una rama

- Donde es importante saber en que rama se desarrolló los commits

2. ¿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?

- Algunas ventajas son: El historial es mas claro y estructurado ya que se puede identificar el trabajo como una feature , bugfix . Facilita el seguimiento de las ramas y tareas que se integraron por temas de auditoría.

- Desventajas : Pueden haber muchos commits de merge si se usan en ramas pequeñas lo que hace que el historial sea un poco ruidoso y complicado de entender .

3. ¿Cuándo es recomendable utilizar una fusión squash? ¿Qué ventajas ofrece para proyectos grandes en comparación con fusiones estándar?

- Cuando se quiere limpiar el historial , combinando todos los commits de una rama en uno solo

- Cuando es más importante el historial principal por temas de fácil lectura y navegación

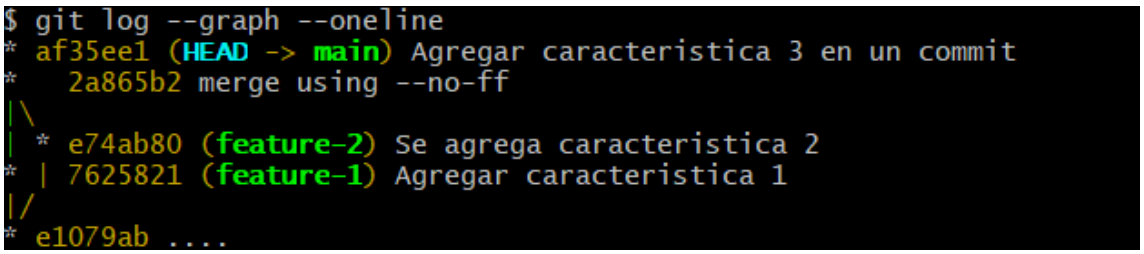
- Se desea integrar la nueva funcionalidad como una unidad lógica sin mostrar los commits intermedios como (test, correcciones , entre otros) .

Resolver conflictos en una fusión non-fast-forward

- ¿Qué pasos adicionales tuviste que tomar para resolver el conflicto?
    - Tuve que editar el archivo index.html y luego realizar el commit
- ¿Qué estrategias podrías emplear para evitar conflictos en futuros desarrollos colaborativos?
    - Mantener la rama actualizada
    - Asignar tareas específicas para cada desarrollador
    - Comunicación continua para coordinar cambios importantes en el código

Comparar las historias con git log después de diferentes fusiones

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



- ¿Qué método prefieres en diferentes escenarios y por qué?
    - Cuando son pocos los commits de la otra rama prefiero usar ff
    - Cuando quiero tener un historial más documentado y ver explícitamente desde que commit se implementaron nuevos cambios uso el noff
    - Cuando veo que los commits de la otra rama son demasiados y además de eso abundan la misma feature , entonces uso squash

Usando fusiones automáticas

- ¿Cuándo usarías un comando como git revert para deshacer una fusión?
    - Cuando hize un commit que provocó un bug y aun no he podido encontrar la solución , y el proyecto tiene que seguir avanzando.
- ¿Qué tan útil es la función de fusión automática en Git?
    - La función de fusión automática de git es muy útil y eficiente porque permite combinar ramas sin intervención manual cuando no hay conflictos , permite ahorrar tiempo , minimiza errores.

Fusión remota en un repositorio colaborativo

- **¿Cómo cambia la estrategia de fusión cuando colaboras con otras personas en un repositorio remoto?**
    - Cuando se trabaja con otros desarrolladores se tiene que ser más organizado y cuidadoso , se recomienda que la rama siempre debe estar actualizada(git pull) , que cuando se va a trabajar en una

funcionalidad entonces se cree una nueva rama donde se desarrolle y pruebe(test) antes de subirlo a la rama de trabajo.

- **¿Qué problemas comunes pueden surgir al integrar ramas remotas?**
    - Conflictos de fusión por temas de manejar una rama desactualizada, edición de un archivo , problemas por la alteración del historial de commits no permitiendo pushear a la rama remota

Flujo de trabajo completo

Rama master :

```
$ git log
commit 43ef84e6020812728fbfd47c322073a5abb7ff3a (HEAD -> master, feature1)
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:55:37 2025 -0500

    Revert "add new line in hola2.py for colaborator2"

    This reverts commit 77889a90807c5419f3e5d21c9407092b4f12e3a2.

commit 77889a90807c5419f3e5d21c9407092b4f12e3a2
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:54:26 2025 -0500

    add new line in hola2.py for colaborator2

commit c3571a287eac6521b9a9980b227197799709e154
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:53:28 2025 -0500

    add new line in hola.py

commit 049977f351ace9be4c97a1c1cdfeea5f7427bcbe
Author: colaborador1 <colaborador1@gmail.com>
Date:   Tue Apr 8 23:51:45 2025 -0500

    first commit
```

Rama feature1:

```
$ git log
commit 43ef84e6020812728fbfd47c322073a5abb7ff3a (HEAD -> feature1, master)
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:55:37 2025 -0500

    Revert "add new line in hola2.py for colaborator2"

    This reverts commit 77889a90807c5419f3e5d21c9407092b4f12e3a2.

commit 77889a90807c5419f3e5d21c9407092b4f12e3a2
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:54:26 2025 -0500

    add new line in hola2.py for colaborator2

commit c3571a287eac6521b9a9980b227197799709e154
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:53:28 2025 -0500

    add new line in hola.py

commit 049977f351ace9be4c97a1c1cdfeea5f7427bcbe
Author: colaborador1 <colaborador1@gmail.com>
Date:   Tue Apr 8 23:51:45 2025 -0500

    first commit
```

Rama feature2:

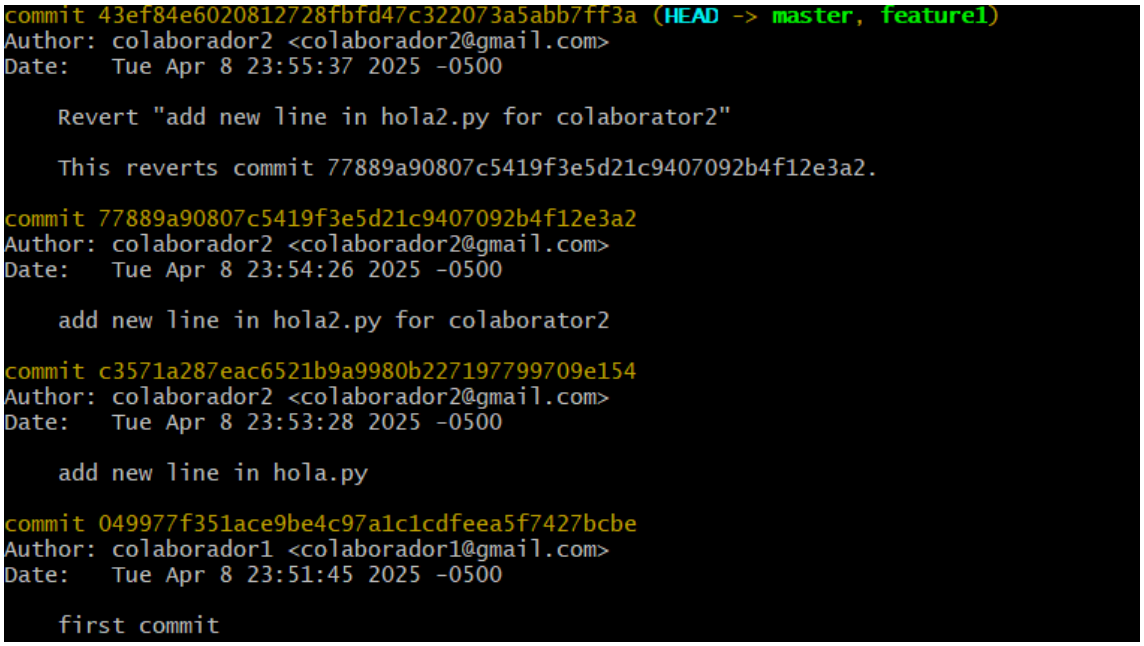
```
$ git log
commit 6ea42cc2278c06d6058560d1e9a263019e174c4f (HEAD -> feature2)
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:00:00 2025 -0500

    add prueba.js

commit 049977f351ace9be4c97a1c1cdfeea5f7427bcbe
Author: colaborador1 <colaborador1@gmail.com>
Date:   Tue Apr 8 23:51:45 2025 -0500

    first commit
```

Después de git merge feature1 –ff



Después de git merge feature2 --no-ff

```
gabri@Gabriel2020 MINGW64 /c/Proyectos/repositorioPrueba/proyecto-simulado (master)
$ git log
commit ed6c9c48321cc27e49f010ba96a384bc44897ae4 (HEAD -> master)
Merge: 43ef84e 249e8ed
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:14:29 2025 -0500

    Merge branch 'feature2' for add saludo

commit 249e8eddb7da439bc4dd9abb9d6fe30ec97d8374 (feature2)
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:13:47 2025 -0500

    add hola3.py

commit 6ea42cc2278c06d6058560d1e9a263019e174c4f
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:00:00 2025 -0500

    add prueba.js

commit 43ef84e6020812728fbfd47c322073a5abb7ff3a (feature1)
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:55:37 2025 -0500

    Revert "add new line in hola2.py for colaborator2"

    This reverts commit 77889a90807c5419f3e5d21c9407092b4f12e3a2.

commit 77889a90807c5419f3e5d21c9407092b4f12e3a2
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:54:26 2025 -0500

    add new line in hola2.py for colaborator2

commit c3571a287eac6521b9a9980b227197799709e154
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:53:28 2025 -0500

    add new line in hola.py

commit 049977f351ace9be4c97a1c1cdfeea5f7427bcbe
Author: colaborador1 <colaborador1@gmail.com>
Date:   Tue Apr 8 23:51:45 2025 -0500

    first commit
```

Rama feature3

```
gabri@Gabriel2020 MINGW64 /c/Proyectos/repositorioPrueba/proyecto-simulado (feature3)
$ git log
commit d5d85b03f9f5e483095b289f9ebd04d20e9fa12b (HEAD -> feature3)
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:20:59 2025 -0500

    add linea2.js

commit 4f5346c9bb94620887a1cd2007eec13b921e3b26
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:20:45 2025 -0500

    add linea1.js
```

Después de git merge --squash feature3

```
gabri@Gabriel2020 MINGW64 /c/Proyectos/repositorioPrueba/proyecto-simulado (master)
$ git log
commit ddbf270870df97303f4975c224da4fd1b0ed3d07 (HEAD -> master)
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:23:03 2025 -0500

    add linea1 and line2

commit ed6c9c48321cc27e49f010ba96a384bc44897ae4
Merge: 43ef84e 249e8ed
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:14:29 2025 -0500

    Merge branch 'feature2' for add saludo

commit 249e8eddb7da439bc4dd9abb9d6fe30ec97d8374 (feature2)
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:13:47 2025 -0500

    add hola3.py

commit 6ea42cc2278c06d6058560d1e9a263019e174c4f
Author: colaborador3 <colaborador3@gmail.com>
Date:   Wed Apr 9 00:00:00 2025 -0500

    add prueba.js

commit 43ef84e6020812728fbfd47c322073a5abb7ff3a (feature1)
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:55:37 2025 -0500

    Revert "add new line in hola2.py for colaborator2"

    This reverts commit 77889a90807c5419f3e5d21c9407092b4f12e3a2.

commit 77889a90807c5419f3e5d21c9407092b4f12e3a2
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:54:26 2025 -0500

    add new line in hola2.py for colaborator2

commit c3571a287eac6521b9a9980b227197799709e154
Author: colaborador2 <colaborador2@gmail.com>
Date:   Tue Apr 8 23:53:28 2025 -0500

    add new line in hola.py

commit 049977f351ace9be4c97a1c1cdfeea5f7427bcbe
Author: colaborador1 <colaborador1@gmail.com>
Date:   Tue Apr 8 23:51:45 2025 -0500

    first commit
```

```
$ git log --graph --oneline
* ddbf270 (HEAD -> master) add linea1 and line2
*   ed6c9c4 Merge branch 'feature2' for add saludo
|\
| * 249e8ed (feature2) add hola3.py
| * 6ea42cc add prueba.js
* | 43ef84e (feature1) Revert "add new line in hola2.py for colaborator2"
* | 77889a9 add new line in hola2.py for colaborator2
* | c3571a2 add new line in hola.py
|/
* 049977f first commit
```

Conclusiones :

- Para proyectos más grandes para cada funcionalidad debería crear ramas y así cuando tienen muchos commits que puedan despistar el historial de commits de la rama principal lo mejor será usar merge squash , si quiero que esté bien documentado podría agregar el –no-ff .