

Trabajo Práctico N°2 Git y GitHub

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas).

- **¿Qué es GitHub?**

Es una plataforma colaborativa que permite al usuario alojar repositorios de GIT fácilmente en la web. Se utiliza mucho en la gestión de proyectos de software y permite colaboración entre distintos desarrolladores/diseñadores.

- **¿Cómo crear un repositorio en GitHub?**

Primero que nada es requisito fundamental crearse una cuenta en la plataforma; una vez registrado iniciar sesión; dentro del listado de repositorios situarse sobre la opción “new” y de esa forma el sistema sabrá que quieres crear un nuevo repositorio; seguido nos genera un formulario el cual solicita datos como nombre, privacidad, y demás configuraciones que debemos aplicar al nuevo repositorio en cuestión.

- **¿Cómo crear una rama en Git?**

Una rama se crea fácilmente escribiendo por consola el comando *git branch nombre-rama* y esta es una copia independiente de la línea principal del desarrollo.

- **¿Cómo cambiar a una rama en Git?**

Para cambiar de rama se escribe en consola el siguiente comando *git checkout nombre-rama*

- **¿Cómo fusionar ramas en Git?**

Para la fusión de ramas en Git es necesario situarse en la rama principal a la que se le quiera fusionar otra rama y escribir el siguiente comando *git merge nombre-rama*

- **¿Cómo crear un commit en Git?**

Para la creación del commit se debe ejecutar el siguiente comando *git commit -m “agregando mi archivo”*

- **¿Cómo enviar un commit a GitHub?**

Para el envío del cambio de repositorio remoto se debe ejecutar el siguiente comando *git push -u origin main*

- **¿Qué es un repositorio remoto?**

Es un repositorio en la nube, virtual, donde permite la colaboración de los desarrolladores y el intercambio de código, un ejemplo más que claro es GitHub.

- **¿Cómo agregar un repositorio remoto a Git?**

Para agregar un repositorio remoto a GIT se debe ejecutar el siguiente comando *git remote add <nombrerepo><URL-Repo>*

- **¿Cómo empujar cambios a un repositorio remoto?**

Para empujar cambios a un repositorio remoto se debe ejecutar el siguiente comando
git push -u origin main

- **¿Cómo tirar de cambios de un repositorio remoto?**

Para traer las actualizaciones del repositorio remoto al local, se debe ejecutar el siguiente comando
git pull <nombre-remoto><nombre-rama>

- **¿Qué es un fork de repositorio?**

Un fork es una copia personal de un repositorio de GitHub y permite realizar cambios sobre el proyecto sin afectar al original.

- **¿Cómo crear un fork de un repositorio?**

Para crear un fork, hay que dirigirse al repositorio original del desarrollador por ejemplo dentro de GitHub; hacer click en el botón “Fork” ubicado a la derecha superior de la página; elegir donde ubicar la creación dentro de mi cuenta y paso seguido se creará el mismo.

- **¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?**

Para enviar una solicitud de pull request primero que nada se debe realizar mis cambios y hacer un commit en una rama separada de mi fork; luego subir la rama y crear la pull request mediante la opción “New pull request”, seleccionar la rama que contiene mis cambios y la rama donde quiero fusionar; redactar un título con una descripción de los cambios efectuados y finalmente presionar en “Create pull request”.

- **¿Cómo aceptar una solicitud de extracción?**

Al aceptar una pull request estamos integrando los cambios propuestos en la rama destino. Para realizar esto debemos principalmente revisar los cambios que nos propone la otra persona, para ver si están acordes al objetivo del proyecto, si es así hacer clic en “Merge pull request” y confirmar la acción.

- **¿Qué es una etiqueta en Git?**

Una etiqueta, o un tag, es una referencia específica sobre los cambios. Se utilizan para marcar puntos de lanzamiento estables, como por ejemplo V1.0, V2-beta.

- **¿Cómo crear una etiqueta en Git?**

Se crean mediante el comando *git tag -a <nombre-etiqueta>*

- **¿Cómo enviar una etiqueta a GitHub?**

Para enviar todas las etiquetas se debe usar el siguiente comando
git push origin --tags

Y para enviar una etiqueta en específica se usa el siguiente comando
git push origin <nombre-etiqueta>

- **¿Qué es un historial de Git?**

Básicamente el historial de Git se refiere al historial de cambios realizados en el repositorio y se organiza en una secuencia de commits. En ellos se pueden ver características como los cambios en si, quien los realizó, en qué momento.

- **¿Cómo ver el historial de Git?**

El historial se puede visualizar mediante el comando *git log*

- **¿Cómo buscar en el historial de Git?**

Para buscar en el historial se pueden agregar parametros como por ejemplo:

git log --author="Franco Reynoso"

- **¿Cómo borrar el historial de Git?**

Para eliminar el historial se debe ejecutar el siguiente comando

git reset

- **¿Qué es un repositorio privado en GitHub?**

Un repositorio privado es aquel en el que solo tiene acceso el propietario, y las cuentas que él invite a colaborar.

- **¿Cómo crear un repositorio privado en GitHub?**

Se debe crear un nuevo repositorio y en la sección de visibilidad elegir "Private".

- **¿Cómo invitar a alguien a un repositorio privado en GitHub?**

Para invitar a un colaborador a un repositorio privado, se debe ir a los ajustes del repositorio en cuestión, ubicarse sobre el panel de Colaboradores y Equipo, ingresar un nombre de usuario/correo electrónico del usuario, asignarle un rol como por ejemplo editor, y finalmente agregarlo mediante el botón add collaborator. A la persona le llegará la invitación y deberá aceptarla para obtener su acceso al repositorio.

- **¿Qué es un repositorio público en GitHub?**

Un repositorio público es lo opuesto al privado, es decir que en estos cualquier persona tiene acceso al mismo, y dependiendo la configuración de este se podrá visualizar, descargar, y en algunos casos contribuir con el proyecto.

- **¿Cómo crear un repositorio público en GitHub?**

Se debe crear un nuevo repositorio y en la sección de visibilidad elegir "Public".

- **¿Cómo compartir un repositorio público en GitHub?**

La forma más sencilla y común es compartir la URL del repositorio.

2) Realizar la siguiente actividad:

- Crear un repositorio.
- Dale un nombre al repositorio.
- Elige que el repositorio sea público.
- Inicializa el repositorio con un archivo.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Required fields are marked with an asterisk ().*

Repository template

No template ▾

Start your repository with a template repository's contents.


Owner * **Repository name ***


 FranVD ▾ /

✓ Your new repository will be created as **programaci-nUTNu2**.
The repository name can only contain ASCII letters, digits, and the characters ., -, and _.

Great repository names are short and memorable. Need inspiration? How about **probable-computing-machine** ?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore


.gitignore template: **None** ▾


Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **None** ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  **main** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

[Create repository](#)

- Agregando un Archivo

- Crea un archivo simple, por ejemplo, "mi-archivo.txt".
- Realiza los comandos `git add .` y `git commit -m "Agregando mi-archivo.txt"` en la línea de comandos.
- Sube los cambios al repositorio en GitHub con `git push origin main` (o el nombre de la rama correspondiente).

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\Franco> cd programaci-nUTNu2
PS C:\Users\Franco\programaci-nUTNu2> echo "Esto es un ejemplo para la actividad número 2 del TP 2 de la materia Programación 1" > mi-archivo.txt
PS C:\Users\Franco\programaci-nUTNu2> dir

Directorio: C:\Users\Franco\programaci-nUTNu2

Mode                LastWriteTime         Length Name
----                -
-a----            20/4/2025   02:54             172 mi-archivo.txt
-a----            20/4/2025   02:48             137 README.md

PS C:\Users\Franco\programaci-nUTNu2> git add .
PS C:\Users\Franco\programaci-nUTNu2> git commit -m "Agregando mi-archivo.txt"
Author identity unknown

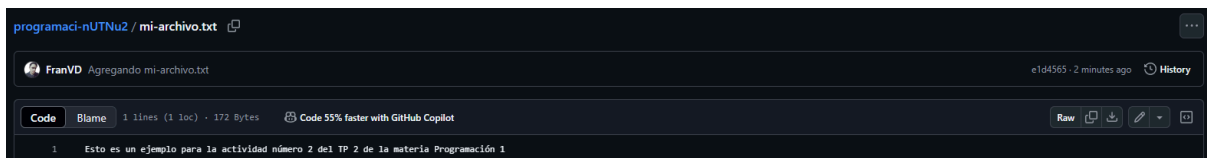
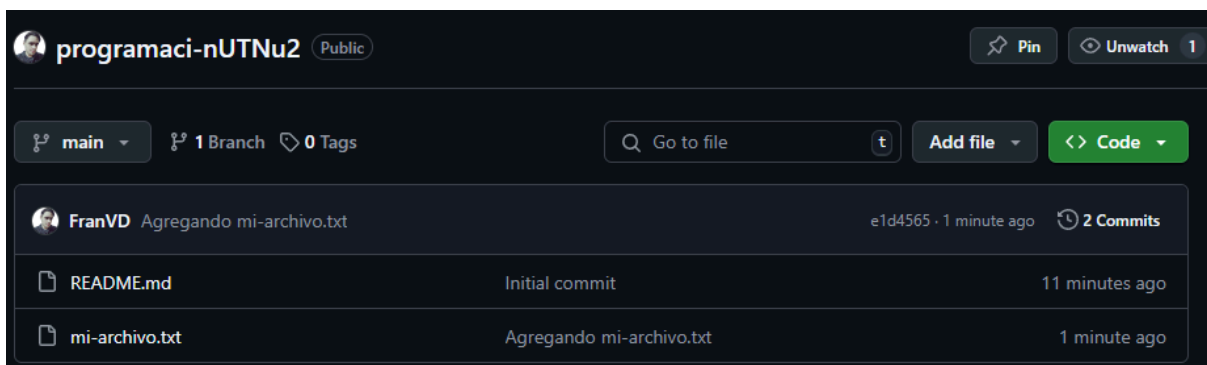
*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

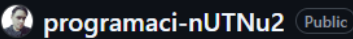
fatal: unable to auto-detect email address (got 'Franco@Franco.(none)')
PS C:\Users\Franco\programaci-nUTNu2> git config --global user.email "franreynosol2@gmail.com"
PS C:\Users\Franco\programaci-nUTNu2> git config --global user.name "FranVD"
PS C:\Users\Franco\programaci-nUTNu2> git commit -m "Agregando mi-archivo.txt"
[main e1d4565] Agregando mi-archivo.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 mi-archivo.txt
PS C:\Users\Franco\programaci-nUTNu2> git push origin main
info: please complete authentication in your browser...
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 404 bytes | 404.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/FranVD/programaci-nUTNu2.git
 c1728e5..e1d4565  main -> main
PS C:\Users\Franco\programaci-nUTNu2> |
```






• Creando Branchs

- Crear una Branch
- Realizar cambios o agregar un archivo
- Subir la Branch

```
PS C:\Users\Franco\programaci-nUTNu2> git branch mi-primer-branch
PS C:\Users\Franco\programaci-nUTNu2> git checkout mi-primer-branch
Switched to branch 'mi-primer-branch'
PS C:\Users\Franco\programaci-nUTNu2> echo "Esto es una linea agregada via branch" >> mi-archivo.txt
PS C:\Users\Franco\programaci-nUTNu2> type mi-archivo.txt
Esto es un ejemplo para la actividad número 2 del TP 2 de la materia Programación 1
Esto es una linea agregada via branch
PS C:\Users\Franco\programaci-nUTNu2> git add mi-archivo.txt
PS C:\Users\Franco\programaci-nUTNu2> git commit -m "Edite via branch en mi-archivo.txt"
[mi-primer-branch f9b4b8c] Edite via branch en mi-archivo.txt
1 file changed, 0 insertions(+), 0 deletions(-)
PS C:\Users\Franco\programaci-nUTNu2> git push -u origin mi-primer-branch
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 16 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 389 bytes | 389.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'mi-primer-branch' on GitHub by visiting:
remote:   https://github.com/FranVD/programaci-nUTNu2/pull/new/mi-primer-branch
remote:
To https://github.com/FranVD/programaci-nUTNu2.git
 * [new branch]      mi-primer-branch -> mi-primer-branch
branch 'mi-primer-branch' set up to track 'origin/mi-primer-branch'.
PS C:\Users\Franco\programaci-nUTNu2>
```



 1

 **mi-primer-branch** had recent pushes 53 seconds ago




Compare & pull request

Branches New branch




Overview **Yours** Active Stale All




Q Search branches...



Default

Branch	Updated	Check status	Behind Ahead	Pull request
main 	 11 minutes ago		Default	 ...


Your branches



Branch	Updated	Check status	Behind Ahead	Pull request
mi-primer-branch 	 1 minute ago		0 1	 ...


 **mi-primer-branch**  2 Branches  0 Tags


Go to file  Add file  Code

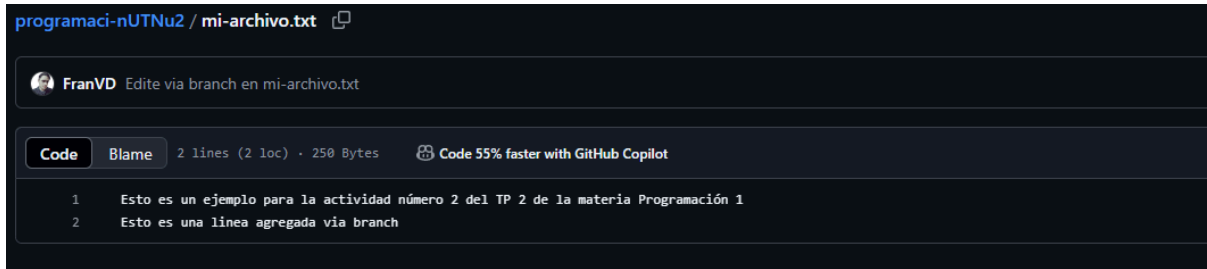
This branch is 1 commit ahead of **main**.

 Contribute

 **FranVD** Edite via branch en mi-archivo.txt f9b4b8c · 2 minutes ago  3 Commits

 README.md Initial commit 22 minutes ago

 mi-archivo.txt Edite via branch en mi-archivo.txt 2 minutes ago



programaci-nUTNu2 / mi-archivo.txt

FranVD Edite via branch en mi-archivo.txt

Code Blame 2 lines (2 loc) · 250 Bytes Code 55% faster with GitHub Copilot

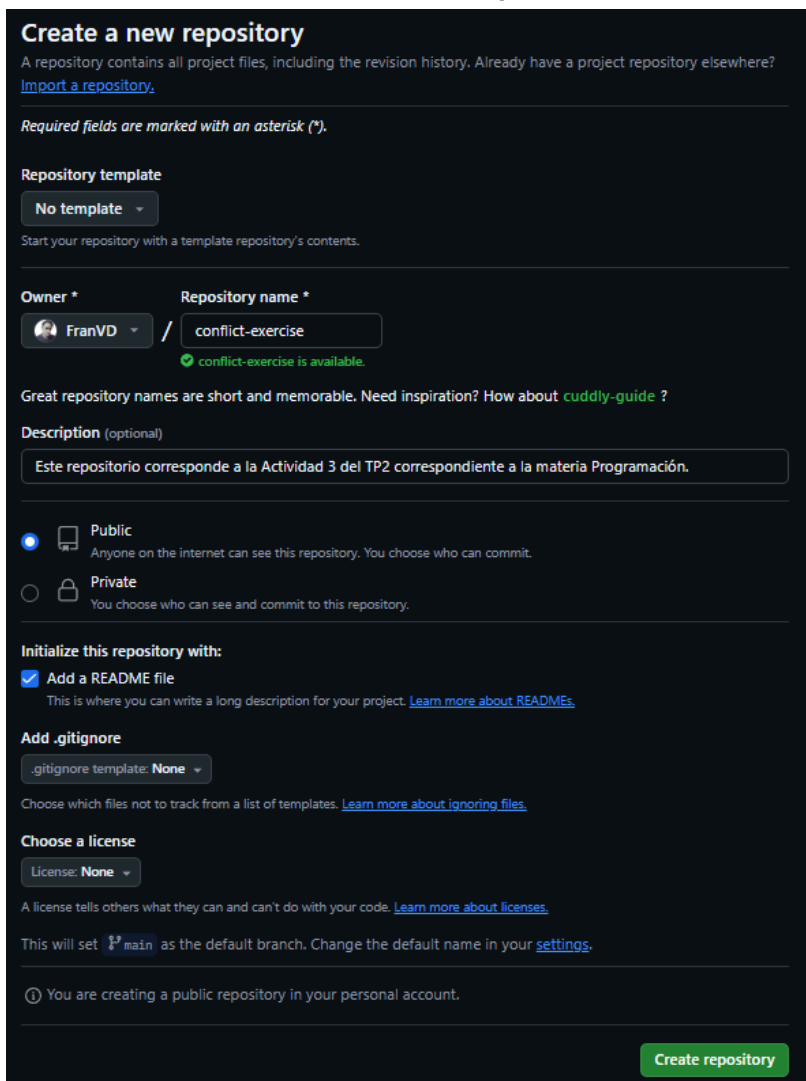
```
1 Esto es un ejemplo para la actividad número 2 del TP 2 de la materia Programación 1
2 Esto es una línea agregada via branch
```

Link al repositorio: <https://github.com/FranVD/programaci-nUTNu2.git>

3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

- Ve a GitHub e inicia sesión en tu cuenta.
- Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
- Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
- Opcionalmente, añade una descripción.
- Marca la opción "Initialize this repository with a README".
- Haz clic en "Create repository"



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).

Repository template

No template

Start your repository with a template repository's contents.

Owner * Repository name *

FranVD / conflict-exercise

conflict-exercise is available.

Great repository names are short and memorable. Need inspiration? How about [cuddly-guide](#) ?

Description (optional)

Este repositorio corresponde a la Actividad 3 del TP2 correspondiente a la materia Programación.

Public

Anyone on the internet can see this repository. You choose who can commit.

Private

You choose who can see and commit to this repository.

Initialize this repository with:

☒ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set `main` as the default branch. Change the default name in your [settings](#).

☐ You are creating a public repository in your personal account.

Create repository

Paso 2: Clonar el repositorio a tu máquina local

- Copia la URL del repositorio
- Abre la terminal o línea de comandos en tu máquina.
- Clona el repositorio
- Entra en el directorio del repositorio

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\Franco> git clone https://github.com/FranVD/conflict-exercise.git
Cloning into 'conflict-exercise'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\Franco> cd conflict-exercise
PS C:\Users\Franco\conflict-exercise> |
```

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch
- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo: Este es un cambio en la feature branch.
- Guarda los cambios y haz un commit

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows

PS C:\Users\Franco> git clone https://github.com/FranVD/conflict-exercise.git
Cloning into 'conflict-exercise'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS C:\Users\Franco> cd conflict-exercise
PS C:\Users\Franco\conflict-exercise> git checkout -b feature-branch
Switched to a new branch 'feature-branch'
PS C:\Users\Franco\conflict-exercise> |
```

```
README.md
Archivo  Editar  Ver

# conflict-exercise
Este repositorio corresponde a la Actividad 3 del TP2 correspondiente a la materia
Programación.

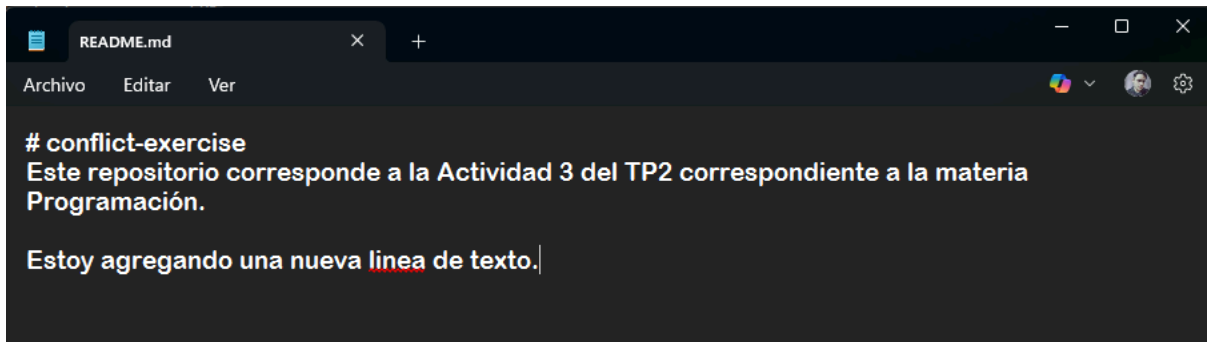
Este es un cambio en la feature branch.

PS C:\Users\Franco\conflict-exercise> git add README.md
PS C:\Users\Franco\conflict-exercise> git commit -m "Added a line in feature-branch"
[feature-branch 950e9a8] Added a line in feature-branch
 1 file changed, 2 insertions(+)
PS C:\Users\Franco\conflict-exercise> |
```


Paso 4: Volver a la rama principal y editar el mismo archivo

- Cambia de vuelta a la rama principal (main):
- Edita el archivo README.md de nuevo, añadiendo una línea diferente
- Guarda los cambios y haz un commit:

```
PS C:\Users\Franco\conflict-exercise> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\Franco\conflict-exercise> |
```



```
PS C:\Users\Franco\conflict-exercise> git add README.md
PS C:\Users\Franco\conflict-exercise> git commit -m "Added a line in main branch"
[main d7dbc8f] Added a line in main branch
1 file changed, 2 insertions(+)
PS C:\Users\Franco\conflict-exercise> |
```

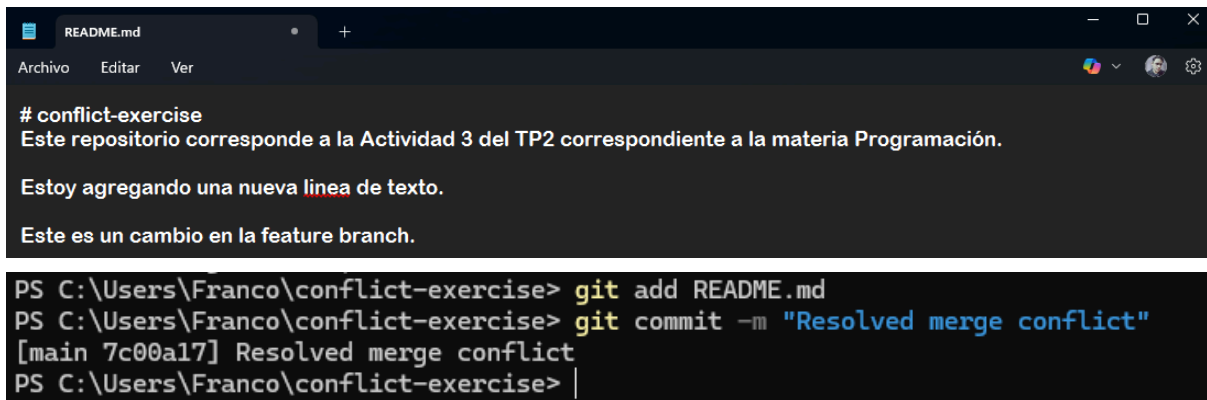
Paso 5: Hacer un merge y generar un conflicto

- Intenta hacer un merge de la feature-branch en la rama main:
- Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

```
PS C:\Users\Franco\conflict-exercise> git merge feature-branch
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.
PS C:\Users\Franco\conflict-exercise> |
```

Paso 6: Resolver el conflicto

- Abre el archivo README.md en tu editor de texto.
- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
- Edita el archivo para resolver el conflicto y guarda los cambios.
- Añade el archivo resuelto y completa el merge



The screenshot shows a code editor window with a tab labeled 'README.md'. The editor contains the following text:

```
# conflict-exercise
Este repositorio corresponde a la Actividad 3 del TP2 correspondiente a la materia Programación.

Estoy agregando una nueva línea de texto.

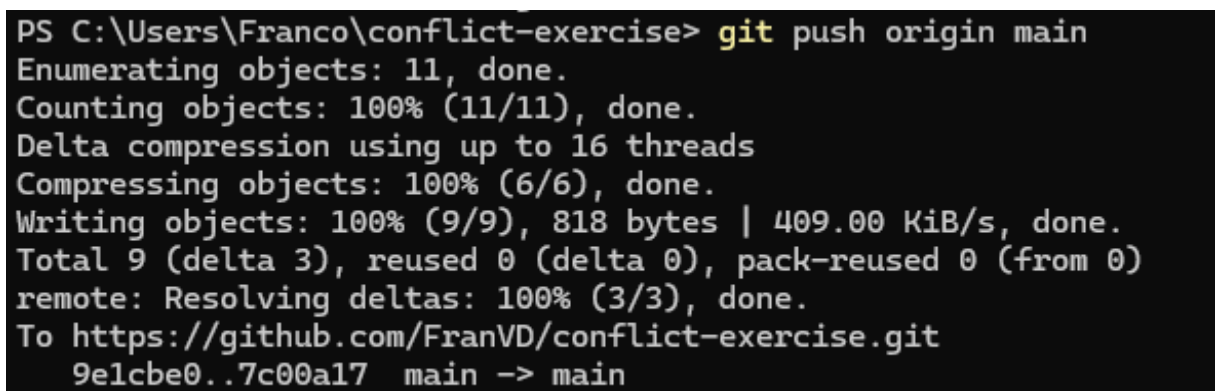
Este es un cambio en la feature branch.
```

Below the editor, a terminal window shows the following commands and output:

```
PS C:\Users\Franco\conflict-exercise> git add README.md
PS C:\Users\Franco\conflict-exercise> git commit -m "Resolved merge conflict"
[main 7c00a17] Resolved merge conflict
PS C:\Users\Franco\conflict-exercise> |
```

Paso 7: Subir los cambios a GitHub

- Sube los cambios de la rama main al repositorio remoto en GitHub
- También sube la feature-branch



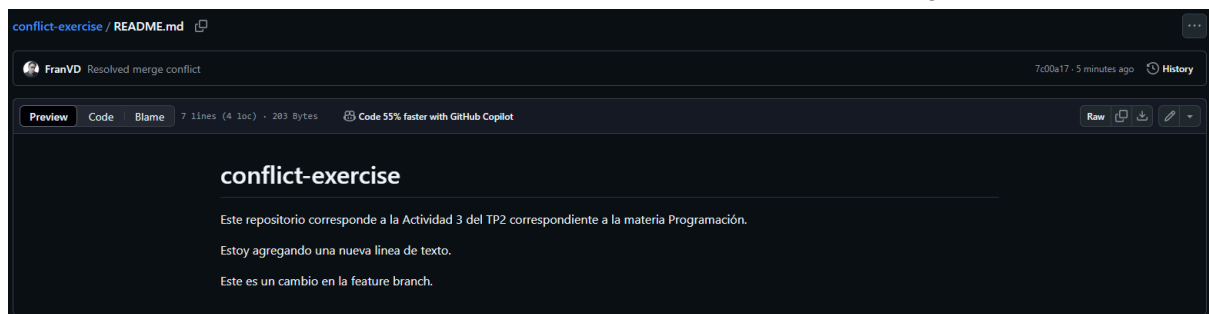
The screenshot shows a terminal window with the following output:

```
PS C:\Users\Franco\conflict-exercise> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 16 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 818 bytes | 409.00 KiB/s, done.
Total 9 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), done.
To https://github.com/FranVD/conflict-exercise.git
  9e1cbe0..7c00a17  main -> main
```

```
PS C:\Users\Franco\conflict-exercise> git push origin feature-branch
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature-branch' on GitHub by visiting:
remote:   https://github.com/FranVD/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/FranVD/conflict-exercise.git
 * [new branch]      feature-branch -> feature-branch
PS C:\Users\Franco\conflict-exercise> |
```

Paso 8: Verificar en GitHub

- Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
- Puedes revisar el historial de commits para ver el conflicto y su resolución.



```
PS C:\Users\Franco\conflict-exercise> git log
commit 7c00a1766ce209fcc1b5da314532bf350f7ee6b9 (HEAD -> main, origin/main, origin/HEAD)
Merge: d7dbc8f 950e9a8
Author: FranVD <franreynoso12@gmail.com>
Date: Sun Apr 20 03:38:15 2025 -0300

    Resolved merge conflict

commit d7dbc8f78780010e2893da35f9656c40f94d44af
Author: FranVD <franreynoso12@gmail.com>
Date: Sun Apr 20 03:31:47 2025 -0300

    Added a line in main branch

commit 950e9a8f8d9beb451c8869ddf94a5d1eee6f89f4 (origin/feature-branch, feature-branch)
Author: FranVD <franreynoso12@gmail.com>
Date: Sun Apr 20 03:27:49 2025 -0300

    Added a line in feature-branch

commit 9e1cbe06bab21fc64a294c708202bcb29b2744aa
Author: Franco Reynoso <141681355+FranVD@users.noreply.github.com>
Date: Sun Apr 20 03:14:21 2025 -0300

    Initial commit
```

LINK DEL REPOSITORIO: <https://github.com/FranVD/conflict-exercise.git>