

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada:

• **¿Qué es GitHub?**

Git es un sistema de control de versiones diseñado que permite rastrear cambios en archivos de un proyecto, y coordinar el trabajo entre varias personas.

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

Una vez instalado GIT, y abierta la consola (Git Bash, CMD, PowerShell o Consola de VSC) en la carpeta principal del proyecto, se inicia el repositorio con el comando 'git init'.

Luego se deben incluir los archivos que se quieren rastrear, con el comando 'git add archivo.py', o 'git add .' para incluir todos los archivos en el repositorio.

Por último, para guardar o "comitear" los cambios, se debe ejecutar el comando: 'git commit -m "Comentario descriptivo" '.

• **¿Cómo crear una rama en Git?**

Al crear un repositorio, por defecto los archivos se guardan en la rama principal llamada 'main' o 'master'.

Se pueden crear ramas o bifurcaciones del proyecto, con el comando 'git brach nuevaRama', dónde nuevaRama será el nombre de dicha rama.

Se pueden consultar las ramas existentes con el comando 'git branch', o 'git branch -a', este último incluye ramas remotas.

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

Para cambiar de rama, se debe ejecutar el comando 'git checkout nuevaRama'.

También puede utilizarse el comando 'git switch nuevaRama'. Este comando es específico para esta acción, a diferencia de 'checkout' que permite otras funcionalidades.

• **¿Cómo fusionar ramas en Git?**

Para fusionar ramas, se ejecuta el comando 'git merge nuevaRama', esto fusiona la rama nuevaRama con la rama actual.

• **¿Cómo crear un commit en Git?**

Para crear un commit se ejecuta el comando 'git commit -m "Comentario descriptivo" '.

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

Nicolás H. VISINTIN – Programación I – TP 2 – GIT – HITHUB

Para sincronizar un repositorio local con uno remoto, se debe ejecutar el comando 'git push', habiendo previamente vinculado el repositorio remoto.

- **¿Qué es un repositorio remoto?**

Un repositorio remoto es una versión del repositorio local, que se aloja en la nube.

Existen varios servicios para hacerlo, por ejemplo GitHub y GitLab.

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

Se deben vincular los repositorios remoto y local.

Esto se realiza desde la consola en local, con el comando 'git remote add origin URL', dónde URL es el link al repositorio remoto, ejemplo: <https://github.com/miusuario/repositorio.git>

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

Una vez vinculados los repositorios, para empujar o subir los cambios del repositorio local, se debe ejecutar el comando 'git push -u origin master' si es la primera vez que se realiza, o simplemente 'git push' luego de la primera vez.

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

Para tirar o bajar los cambios del repositorio remoto al local, se debe ejecutar el comando 'git pull origin master' si es la primera vez, o simplemente 'git push' si previamente ejecutamos 'git push -u origin master'.

- **¿Qué es un fork de repositorio?**

Un fork es la acción de copiar un repositorio de terceros, en nuestra cuenta GitHub.

Esto permite comenzar a trabajar sobre una copia del repositorio del tercero, que será totalmente independiente.

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

Se debe ingresar al repositorio deseado, y clicar sobre la acción "fork".

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

Una solicitud de extracción o pull request a un repositorio es el proceso con el cuál se proponen cambios al repositorio original en base a los cambios realizados en mi repositorio, para que sean revisados e incluidos en el repositorio original.

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

Nicolás H. VISINTIN – Programación I – TP 2 – GIT – HITHUB

Las solicitudes se visualizan ingresando a la pestaña 'Pull Requests' del repositorio en GitHub.

- ¿Qué es un etiqueta en Git?

Una etiqueta en Git es un marcador que se utiliza para identificar puntos específicos en el historial de un repositorio.

- ¿Cómo crear una etiqueta en Git?

Las etiquetas se crean con el comando `'git tag nombreEtiqueta'`.

Se le puede agregar un mensaje al tag, con el comando `'git tag -a nombreEtiqueta -m mensaje'`.

- ¿Cómo enviar una etiqueta a GitHub?

Por defecto las etiquetas no se envían al repositorio remoto.

Para enviarlas se debe ejecutar el comando 'git push origin nombreEtiqueta' o 'git push origin -tags' para enviarlas a todas juntas.

- ¿Qué es un historial de Git?

Es un registro completo de todos los cambios realizados en un repositorio.

- **¿Cómo ver el historial de Git?**

Para consultar el historia, se ejecuta el comando 'git log'.

- ¿Cómo buscar en el historial de Git?

Se puede buscar por palabras clave del commit con `'git log --grep="palabra clave"'`.

Se puede buscar por nombre del autor del commit con `'git log --author="nombre del autor"'`.

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

Eliminar el historial implica eliminar el repositorio.

Se realiza eliminando la carpeta '.git' o con el comando 'rm -rf .git'.

- ¿Qué es un repositorio privado en GitHub?

Un repositorio privado es aquel al cuál sólo tiene acceso el propietario de la cuenta dónde está guardado el repositorio.

• ¿Cómo crear un repositorio privado en GitHub?

Al crear el repositorio en Gith, se puede elegir que el repositorio sea privado:

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 *

Nvis88

Repository name *

prueba

prueba is available.

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

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.

• ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Desde el repositorio en GitHub, pestaña Settings, menú Collaborators se puede otorgar acceso a terceros, con su nombre de usuario, mail o nombre completo.

• ¿Qué es un repositorio público en GitHub?

Un repositorio público es aquel al que cualquier usuario puede acceder para verlo, o copiarlo (fork).

• ¿Cómo crear un repositorio público en GitHub?

Al crear el respositorio, seleccionado que su visibilidad sea Pública.

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 *

Nvis88

Repository name *

prueba

prueba is available.

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

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.

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

Desde la pestaña Settings del repositorio, menú Code, se puede copiar el enlace al repositorio para compartirlo, siempre que sea público.

2) Realizar la siguiente actividad:

Link repositorio: https://github.com/Nvis88/repositorio_ejercicio2.git

- Crear un repositorio.
- Dale un nombre al repositorio.
- Elije el repositorio sea público.

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 *

Nvis88 ▾

Repository name *

repositorio_ejercicio2

✓ repositorio_ejercicio2 is available.

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

Description (optional)

Programación I - TP 2 - Ejercicio 2

☒



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.

- Inicializa el repositorio con un archivo.

```
MINGW64:/e/xampp/htdocs/UTN/repositorio_ejercicio2
nicol@Depto MINGW64 /e/xampp/htdocs/UTN
$ git clone https://github.com/Nvis88/repositorio_ejercicio2.git
Cloning into 'repositorio_ejercicio2'...
warning: You appear to have cloned an empty repository.
nicol@Depto MINGW64 /e/xampp/htdocs/UTN
$ cd repositorio_ejercicio2
nicol@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ |
```

- Agregando un Archivo o Crea un archivo simple, por ejemplo, "mi-archivo.txt".

```
MINGW64:/e/xampp/htdocs/UTN/repositorio_ejercicio2
nicol@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ touch mi-archivo.txt
nicol@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ echo "Este es un texto de ejemplo" > mi-archivo.txt
nicol@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ |
```

Nicolás H. VISINTIN – Programación I – TP 2 – GIT – HITHUB

- Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.

```
MINGW64/e/xampp/htdocs/UTN/repositorio_ejercicio2
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ git add .
warning: in the working copy of 'mi-archivo.txt', LF will be replaced by CRLF the next time Git touches it
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ git commit -m "Agregando mi-archivo.txt"
[main (root-commit) 6c818a9] Agregando mi-archivo.txt
1 file changed, 1 insertion(+)
create mode 100644 mi-archivo.txt
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$
```

- Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

```
MINGW64/e/xampp/htdocs/UTN/repositorio_ejercicio2
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 260 bytes | 260.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Nvis88/repositorio_ejercicio2.git
 * [new branch]      main -> main
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$
```

Creando Branchs

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

```
MINGW64/e/xampp/htdocs/UTN/repositorio_ejercicio2
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ git branch secundario

nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (main)
$ git checkout secundario
Switched to branch 'secundario'

nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (secundario)
$ touch segundo_archivo.txt

nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (secundario)
$ echo "texto de prueba en segundo archivo" > segundo_archivo.txt

nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (secundario)
$ git add .
warning: in the working copy of 'segundo_archivo.txt', LF will be replaced by CRLF the next time Git touches it

nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (secundario)
$ ^C

nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (secundario)
$ git commit -m "Inicio de rama secundaria"
[secundario d4a4262] Inicio de rama secundaria
1 file changed, 1 insertion(+)
create mode 100644 segundo_archivo.txt
```

Nicolás H. VISINTIN – Programación I – TP 2 – GIT – HITHUB

```
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (secundario)
$ git push -u origin secundario
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (7/7), 779 bytes | 389.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'secundario' on GitHub by visiting:
remote:   https://github.com/Nvis88/repositorio_ejercicio2/pull/new/secundari
o
remote:
To https://github.com/Nvis88/repositorio_ejercicio2.git
 * [new branch]      secundario -> secundario
branch 'secundario' set up to track 'origin/secundario'.

nico1@Depto MINGW64 /e/xampp/htdocs/UTN/repositorio_ejercicio2 (secundario)
$ |
```


3) Realizar la siguiente actividad:

Link repositorio: <https://github.com/Nvis88/conflict-exercise.git>

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 *

Nvis88

Repository name *

conflict-exercise-

conflict-exercise- is available.

Great repository names are short and memorable. Need inspiration? How about [turbo-palm-tree](#) ?

Description (optional)

Programación I - TP 2 - Ejercicio 3

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.](#)

Paso 2: Clonar el repositorio a tu máquina local

- Copia la URL del repositorio: <https://github.com/Nvis88/conflict-exercise.git>
- Abre la terminal o línea de comandos en tu máquina.
- Clona el repositorio usando el comando: `git clone https://github.com/Nvis88/conflict-exercise.git`
- Entra en el directorio del repositorio: `cd conflict-exercise`

```
MINGW64/e:/xampp/htdocs/UTN/conflict-exercise
nicol@Depto MINGW64 /e:/xampp/htdocs/UTN
$ git clone https://github.com/Nvis88/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.

nicol@Depto MINGW64 /e:/xampp/htdocs/UTN
$ cd conflict-exercise

nicol@Depto MINGW64 /e:/xampp/htdocs/UTN/conflict-exercise (main)
$ |
```

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch: `git checkout -b feature-branch`

Nicolás H. VISINTIN – Programación I – TP 2 – GIT – HITHUB

```
MINGW64:/e/xampp/htdocs/UTN/conflict-exercise
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (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:

```
git add README.md
```

```
git commit -m "Added a line in feature-branch"
```

```
MINGW64:/e/xampp/htdocs/UTN/conflict-exercise
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (feature-branch)
$ git add README.md
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (feature-branch)
$ git commit -m "Se agregó línea a Readme.md"
[feature-branch 3c1bf19] Se agregó línea a Readme.md
1 file changed, 2 insertions(+)
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (feature-branch)
$ |
```

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

- Cambia de vuelta a la rama principal (main): git checkout main
- Edita el archivo README.md de nuevo, añadiendo una línea diferente:

Este es un cambio en la main branch.

- Guarda los cambios y haz un commit:

```
git add README.md
```

```
git commit -m "Added a line in main branch"
```

```
MINGW64:/e/xampp/htdocs/UTN/conflict-exercise
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (feature-branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (main)
$ git add README.md
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (main)
$ git commit -m "Modificación Readme.md en la main branch"
[main d45e579] Modificación Readme.md en la main branch
1 file changed, 2 insertions(+)
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (main)
$ |
```

Paso 5: Hacer un merge y generar un conflicto

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

```
MINGW64:/e/xampp/htdocs/UTN/conflict-exercise
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (main)
$ 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.
nico1@Depto MINGW64 /e/xampp/htdocs/UTN/conflict-exercise (main|MERGING)
$ |
```

Paso 6: Resolver el conflicto

- Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

<<<<<<< HEAD

Este es un cambio en la main branch.

=====

Este es un cambio en la feature branch.

>>>>>>> feature-branch

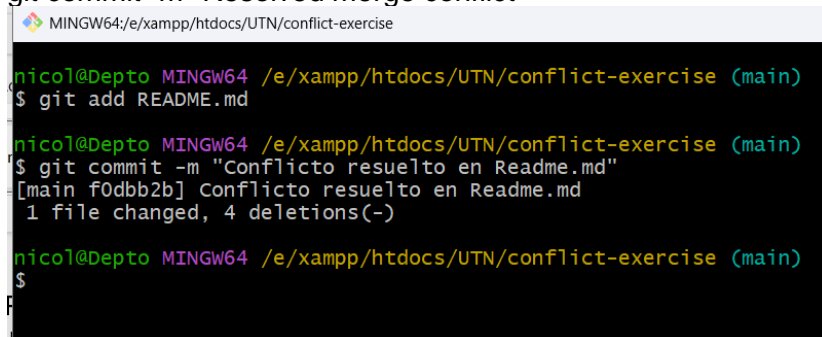
- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.

- Edita el archivo para resolver el conflicto y guarda los cambios (Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).

- Añade el archivo resuelto y completa el merge:

```
git add README.md
```

```
git commit -m "Resolved merge conflict"
```



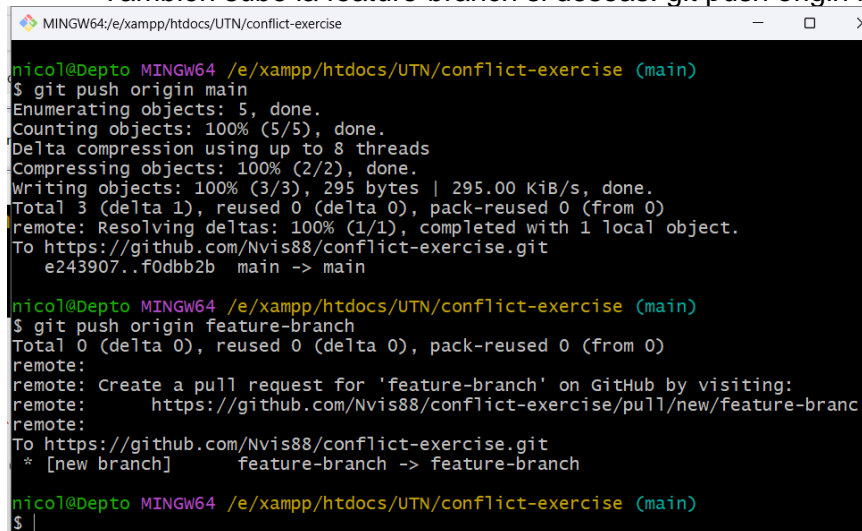
```
MINGW64/e:/xampp/htdocs/UTN/conflict-exercise

nico1@Depto MINGW64 /e:/xampp/htdocs/UTN/conflict-exercise (main)
$ git add README.md

nico1@Depto MINGW64 /e:/xampp/htdocs/UTN/conflict-exercise (main)
$ git commit -m "Conflicto resuelto en Readme.md"
[main f0dbb2b] Conflicto resuelto en Readme.md
1 file changed, 4 deletions(-)
```

Paso 7: Subir los cambios a GitHub

- Sube los cambios de la rama main al repositorio remoto en GitHub: git push origin main
- También sube la feature-branch si deseas: git push origin feature-branch



```
MINGW64/e:/xampp/htdocs/UTN/conflict-exercise

nico1@Depto MINGW64 /e:/xampp/htdocs/UTN/conflict-exercise (main)
$ git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 295 bytes | 295.00 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.
To https://github.com/Nvis88/conflict-exercise.git
e243907..f0dbb2b main -> main


nico1@Depto MINGW64 /e:/xampp/htdocs/UTN/conflict-exercise (main)
$ 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/Nvis88/conflict-exercise/pull/new/feature-branc
remote:
To https://github.com/Nvis88/conflict-exercise.git
* [new branch] feature-branch -> feature-branch



nico1@Depto MINGW64 /e:/xampp/htdocs/UTN/conflict-exercise (main)
$
```


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.


Nicolás H. VISINTIN – Programación I – TP 2 – GIT – HITHUB


 **conflict-exercise** Public


 Pin  Unwatch 1


 main

 2 Branches


 0 Tags


 Add file


 Code

 Nvis88

Conflicto resuelto en Readme.md


 f0dbb2b · 1 minute ago


 5 Commits

 README.md

Conflicto resuelto en Readme.md

1 minute ago


 README



conflict-exercise



Programación I - TP 2 - Ejercicio 3

Este es un cambio en la main Branch. Para generar un conflicto.



1 file changed +0 -4 lines changed

Collapse file tree

 README.md 

...	@@ -1,8 +1,4 @@
1	1 # conflict-exercise
2	2 Programación I - TP 2 - Ejercicio 3
3	3
4	- <<<<<< HEAD
5	4 Este es un cambio en la main Branch. Para generar un conflicto.
6	- =====
7	- Este es un cambio en la feature branch.
8	- >>>>>> feature-branch