

UNIVERSIDAD TECNOLÓGICA NACIONAL

TECNICATURA UNIVERSITARIA EN PROGRAMACIÓN
A DISTANCIA

Práctico 2: Git y GitHub

PROGRAMACIÓN 1

Alumno: Atilano Roberto FERNÁNDEZ

1C2025

Actividades

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

- ¿Qué es GitHub?

***GitHub** es una plataforma en la nube construida alrededor de **Git** (que es un sistema de control de versiones local). Esta plataforma es utilizada por millones de desarrolladores y equipos de trabajo donde alojan, gestionan y colaboran en proyectos de software, documentos, scripts, y más.*

- ¿Cómo crear un repositorio en GitHub?

Para crear un repositorio en GitHub se procede de la siguiente manera:

1. Se **inicia** una sesión en GitHub.
2. Hacer clic en "+" (en la esquina superior derecha) > **"New repository"**.
3. Completar los campos:
 - **Repository name:** Nombre del repositorio (ej: mi-proyecto).
 - **Description:** Descripción opcional.
 - **Public/Private:** Elegir la visibilidad.
 - Iniciar el repositorio con el comando:
Add a README file: Recomendado para empezar.
4. Hacer clic en **"Create repository"**.

- ¿Cómo crear una rama en Git?

Para crear una rama en Git, utilizamos el siguiente comando:

```
git branch <nombre-de-la-rama>
```

- ¿Cómo cambiar a una rama en Git?

Para cambiar una rama en Git, utilizamos el siguiente comando:

```
git checkout -b <nombre-de-la-rama>
```

- ¿Cómo fusionar ramas en Git?

Para fusionar dos ramas en Git, nos posicionamos en la rama destino con el comando checkout y luego utilizamos el siguiente comando para fusionarlas:

```
git merge <nombre-de-la-rama>
```

- ¿Cómo crear un commit en Git?

Para enviar un commit en GitHub, procedemos de la siguiente forma:

Preparar los cambios

- *Mueve el archivo que quieres cargar al directorio local del repositorio.*
- *En la terminal, ejecuta `git add .` para incluir todos los archivos o `git <nombre de archivo>` para incluir uno específico, y hacer un seguimiento de los cambios.*
- *Ejecuta `git commit -m "mensaje"` para preparar los cambios para ser enviados a GitHub.*

- ¿Cómo enviar un commit a GitHub?

Para enviar los cambios realizados en git a GitHub hacemos lo siguiente:

- *Si es la primera vez, hay que vincular el repositorio local con el remoto:*

```
git remote add origin  
https://github.com/<nombre_de_usuario>/<nombre_del_repositorio.git>
```
- *Ahora se ejecuta el comando:*

```
git push -u origin <nombre_de_la_rama> # -u: establece la rama remota por defecto.
```

para enviar los archivos a GitHub.
- *Para futuras actualizaciones, ejecutar solamente:*

```
git push
```

- ¿Qué es un repositorio remoto?

Un repositorio remoto en Git es una copia de un proyecto que se encuentra en una red o en internet. Se puede acceder a él desde cualquier lugar con conexión a internet.

Los repositorios remotos son útiles para colaborar en proyectos colaborativos, ya que permiten compartir y sincronizar cambios entre los miembros de un equipo.

Características de los repositorios remotos:

- *Se pueden tener varios repositorios remotos, cada uno con diferentes permisos.*
- *Se pueden clonar repositorios remotos para crear una copia local en el equipo.*
- *Se pueden administrar repositorios remotos, incluyendo agregar, eliminar, y definir si se rastrean o no.*
- *Se pueden enviar y recibir datos entre repositorios remotos.*

Diferencia entre repositorios locales y remotos:

- *Los repositorios locales se encuentran en un equipo local, mientras que los remotos se encuentran en un servidor remoto.*
- *Los repositorios locales se usan para desarrollar y hacer cambios, mientras que los remotos se usan para colaborar.*

- ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git, se usa el comando:

```
git remote add
```

Se debe estar dentro del directorio donde está almacenado el repositorio.

Paso 1

Usar el comando

```
git remote add origin
```

o

```
git remote add <nombre_remoto> <URL_del_repositorio>
```

También se puede clonar un repositorio remoto usando el comando git clone. Para ello, se necesita la URL del repositorio remoto.

Paso 2

1. Ejecutar git clone.
2. Se extraerá la última versión de los archivos del repositorio remoto en la rama principal.
3. Se añadirá a una nueva carpeta con el mismo nombre que el repositorio

- ¿Cómo empujar cambios a un repositorio remoto?

Usar el comando:

```
git push origin main
```

o

```
git push <nombre_remoto> <nombre_de_la_rama>
```

- ¿Cómo tirar de cambios de un repositorio remoto?

Usar el comando:

```
git pull origin main
```

o

```
git pull <nombre_remoto> <nombre_de_la_rama>
```

- ¿Qué es un fork de repositorio?

*Es una copia de un repositorio de otro usuario en **nuestra cuenta de GitHub**, que permite:*

- *Modificar el proyecto sin afectar el original.*
- *Contribuir al proyecto original mediante Pull Requests.*

- ¿Cómo crear un fork de un repositorio?

1. Ir al repositorio en GitHub que se quiere copiar.
2. Hacer clic en **"Fork"** (esquina superior derecha).
3. Elegir una cuenta como destino.
4. Si es necesario, nombrar y describir el fork
5. Hacer clic en Create fork

Se hará una copia en:

`github.com/<usuario>/<nombre-del-repositorio>`

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
Para enviar una solicitud de extracción (pull request) a un repositorio en GitHub, se deben seguir estos pasos:

1. Ir a la página principal del repositorio
2. Seleccionar la rama que contiene los cambios en el menú "Rama"
3. En el banner amarillo, hacer clic en "Comparar y solicitud de incorporación de cambios"
4. Elegir la rama base y la rama de comparación
5. Escribir un título y una descripción
6. Hacer clic en "Crear solicitud de incorporación de cambios"

- ¿Cómo aceptar una solicitud de extracción?

Pasos (como administrador del repositorio):

1. Revisar los cambios en la pestaña "Files changed".
2. Si hay conflictos, pedir al colaborador que los resuelva.
3. Escribir los comentarios o "Review" para sugerir mejoras.
4. Si todo está listo, hacer clic en "Merge pull request".

- ¿Qué es una etiqueta en Git?

Una etiqueta es una referencia estática que marca un punto específico (a un commit específico) en el historial de un repositorio.

Las etiquetas son útiles para identificar confirmaciones importantes, como las que marcan el inicio de una nueva versión (ej: v1.0.0).

A diferencia de las ramas, las etiquetas no cambian.

- ¿Cómo crear una etiqueta en Git?

Una etiqueta se crea con el siguiente comando:

```
git tag -a v1.0.0 -m "Versión 1.0.0" # Etiqueta anotada (recomendada).  
git tag v1.0.0-light                # Etiqueta ligera (sin mensaje).
```

- ¿Cómo enviar una etiqueta a GitHub?

Una etiqueta se crea con el siguiente comando:

```
git push origin v1.0.0          # Envía una etiqueta específica.  
git push origin --tags          # Envía todas las etiquetas nuevas.
```

- ¿Qué es un historial de Git?

Es el registro de todos los commits realizados en un repositorio, mostrando quién hizo qué cambios y cuándo.

- ¿Cómo ver el historial de Git?

```
git log                # Muestra commits con detalles.
git log --oneline      # Versión resumida (1 línea por
commit).
git log --graph        # Muestra ramas y fusiones
visualmente.
```

- ¿Cómo buscar en el historial de Git?

*Para buscar en el historial de Git, puede usar los comandos **git log** y **git grep**.*

git log

- *Muestra el historial de commits de un archivo*
- *Permite personalizar la salida con opciones como **--stat**, **--name-status**, **--graph**, **--author**, y más.*
- *Para buscar confirmaciones que agreguen o eliminen una cadena específica, usa **git log -p -S <cadena>***
- *Para buscar un patrón en los mensajes de confirmación, usa **git log --grep=<pattern>***

git grep

- *Permite buscar a través de cualquier árbol o directorio de trabajo con commit por una cadena o expresión regular*
- *Es rápido y permite buscar a través de cualquier árbol en Git*
- *Tiene opciones como **-n**, **--count**, **-p**, **--and**, **--break**, y **--heading***

- ¿Cómo borrar el historial de Git?

Si se quiere eliminar todo el historial y comenzar con un único commit inicial (por ejemplo, para un proyecto nuevo):

a. Crear una rama huérfana:

```
git checkout --orphan nueva-rama
git add -A # Agrega todos los archivos actuales
git commit -m "Commit inicial"
```

b. Eliminar la rama principal (por ejemplo, main):

```
git branch -D main # Elimina la rama principal local
git branch -m main # Renombra la rama actual a main
```

c. Forzar un push al repositorio remoto:

```
git push origin --force --all
```

Si solo se necesita borrar commits recientes (ej.: últimos 3 commits):

```
git reset --hard HEAD~3 # Borra los últimos 3 commits
localmente
git push --force # Fuerza el push al remoto
```

- ¿Qué es un repositorio privado en GitHub?

Es un repositorio solo visible para colaboradores designados. Ideal para proyectos confidenciales o personales.

- ¿Cómo crear un repositorio privado en GitHub?

1. Haz clic en "+" > **"New repository"**.
2. En **"Visibility"**, selecciona **"Private"**.
3. Completar los campos y hacer clic en **"Create repository"**.

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

1. Ir al repositorio en GitHub.
2. Hacer clic en **"Settings"** > **"Collaborators"**.
3. En **"Add people"**, escribir el nombre de usuario o email del colaborador.
4. Hacer clic en **"Add"** para enviar la invitación.

- ¿Qué es un repositorio público en GitHub?

*Es un repositorio visible para cualquier persona, incluso sin cuenta de GitHub. Cualquiera puede clonarlo o hacer **fork**, pero solo los colaboradores pueden modificarlo directamente.*

- ¿Cómo crear un repositorio público en GitHub?

1. Haz clic en "+" > **"New repository"**.
2. En **"Visibility"**, selecciona **"Public"**.
3. Completar los campos y hacer clic en **"Create repository"**.

- ¿Cómo compartir un repositorio público en GitHub?

A través de un enlace directo: Copiar la URL del repositorio (ej: https://github.com/<nombre_de_usuario>/<repositorio>).

README.md: Añade una descripción clara para que otros entiendan el proyecto.

*Redes sociales: Usa el botón **"Share"** en GitHub para compartir en Twitter, LinkedIn, etc.*

2) Realizar la siguiente actividad:

- Crear un repositorio.

```
roberto@MacBook-Pro-de-Atilano ~ % cd Desktop/SCRIPTS
roberto@MacBook-Pro-de-Atilano SCRIPTS % git init TP2_repositorio
Reinitialized existing Git repository in /Users/roberto/Desktop/SCRIPTS/TP2_repositorio/.git/
roberto@MacBook-Pro-de-Atilano SCRIPTS %
```

- Agregando un Archivo

```
roberto@MacBook-Pro-de-Atilano SCRIPTS % cd TP2_repositorio
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git add .
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git commit -m "agregamos el archivo mi_archivo_1.txt"
[main (root-commit) bd61ca5] agregamos el archivo mi_archivo_1.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 mi_archivo_1.txt
roberto@MacBook-Pro-de-Atilano TP2_repositorio %
```

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

```
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git remote add origin https://github.com/arfernandez7691/TP2_repositorio.git
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 248 bytes | 248.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/arfernandez7691/TP2_repositorio.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
roberto@MacBook-Pro-de-Atilano TP2_repositorio %
```

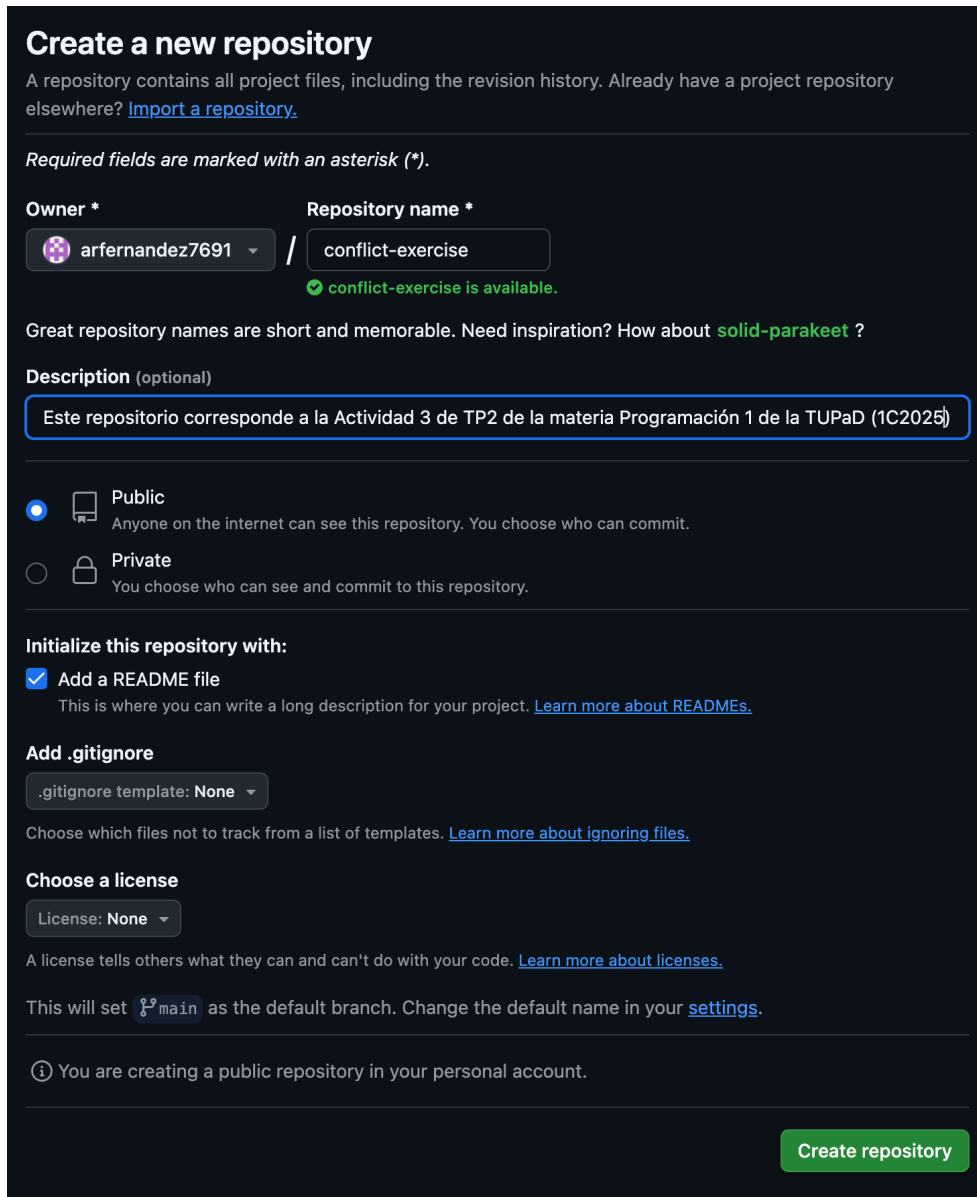
- Creando Branchs

```
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git branch rama_1
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git branch
* main
  rama_1
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git checkout rama_1
Switched to branch 'rama_1'
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git branch
main
  rama_1
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git add .
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git status
On branch rama_1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   mi_archivo_2.txt

roberto@MacBook-Pro-de-Atilano TP2_repositorio % git commit -m "creamos una rama (rama_1) y agregamos el archivo mi_archivo_2.txt a la misma"
[rama_1 b0f381f] creamos una rama (rama_1) y agregamos el archivo mi_archivo_2.txt a la misma
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 mi_archivo_2.txt
roberto@MacBook-Pro-de-Atilano TP2_repositorio % git push -u origin main
branch 'main' set up to track 'origin/main'.
Everything up-to-date
roberto@MacBook-Pro-de-Atilano TP2_repositorio %
```


3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub



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 (*).

Owner * arfernandez7691 / Repository name * conflict-exercise

✔ conflict-exercise is available.

Great repository names are short and memorable. Need inspiration? How about **solid-parakeet** ?

Description (optional)

Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD (1C2025)

☒ 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

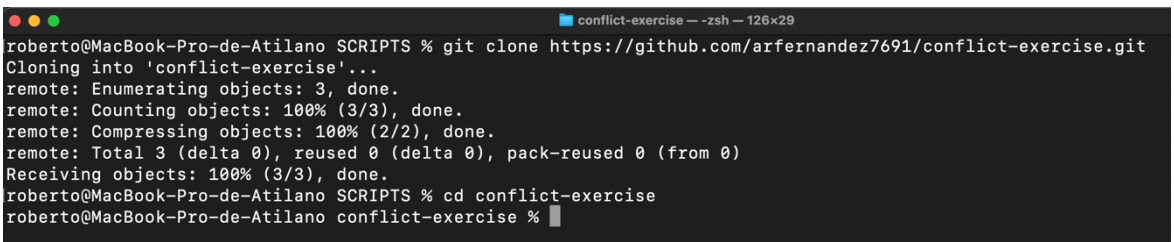
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

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

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

Create repository

Paso 2: Clonar el repositorio a tu máquina local



```
roberto@MacBook-Pro-de-Atilano SCRIPTS % git clone https://github.com/arfernandez7691/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.
roberto@MacBook-Pro-de-Atilano SCRIPTS % cd conflict-exercise
roberto@MacBook-Pro-de-Atilano conflict-exercise %
```

Paso 3: Crear una nueva rama y editar un archivo

```
roberto@MacBook-Pro-de-Atilano SCRIPTS % git clone https://github.com/arfernandez7691/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.
roberto@MacBook-Pro-de-Atilano SCRIPTS % cd conflict-exercise
roberto@MacBook-Pro-de-Atilano conflict-exercise % git checkout -b feature-branch
Switched to a new branch 'feature-branch'
roberto@MacBook-Pro-de-Atilano conflict-exercise % git add README.md
roberto@MacBook-Pro-de-Atilano conflict-exercise % git branch
* feature-branch
  main
roberto@MacBook-Pro-de-Atilano conflict-exercise % git commit -m "Se agregó una línea en feature-branch"
[feature-branch e24b195] Se agregó una línea en feature-branch
1 file changed, 2 insertions(+)
roberto@MacBook-Pro-de-Atilano conflict-exercise % git branch
* feature-branch
  main
roberto@MacBook-Pro-de-Atilano conflict-exercise %
```

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

Rama principal:

```
README.md
# conflict-exercise
Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD (1C2025)
```

Rama feature-branch

```
README.md
# conflict-exercise
Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD (1C2025)

Este es un cambio en la rama "feature-branch".
```

Se editó el archivo README.md en la rama main de nuevo, añadiendo una línea diferente:

```
README.md
# conflict-exercise
Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD (1C2025)

Esta es una línea nueva agregada al archivo en la rama main.
```

Paso 5: Hacer un merge y generar un conflicto

```
roberto@MacBook-Pro-de-Atilano 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.
roberto@MacBook-Pro-de-Atilano conflict-exercise %
```

Paso 6: Resolver el conflicto

```

# conflict-exercise
Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD
(1C2025)

<<<<<< HEAD
Esta es una línea nueva agregada al archivo en la rama main.
=====
Este es un cambio en la rama "feature-branch".
>>>>>> feature-branch
```

Se resuelve el conflicto y el archivo queda:

```

# conflict-exercise
Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD
(1C2025)

Esta es una línea nueva agregada al archivo en la rama main.
```

Paso 7: Subir los cambios a GitHub

```
roberto@MacBook-Pro-de-Atilano 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), 847 bytes | 847.00 KiB/s, done.
Total 9 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To https://github.com/arfernandez7691/conflict-exercise.git
 4d5b7eb..2163022  main -> main
roberto@MacBook-Pro-de-Atilano conflict-exercise %
```

También sube la feature-branch si deseas:

```
roberto@MacBook-Pro-de-Atilano conflict-exercise % git push origin feature-branch
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature-branch' on GitHub by visiting:
remote:   https://github.com/arfernandez7691/conflict-exercise/pull/new/feature-branch
remote:
To https://github.com/arfernandez7691/conflict-exercise.git
 * [new branch]   feature-branch -> feature-branch
roberto@MacBook-Pro-de-Atilano conflict-exercise %
```

Paso 8: Verificar en GitHub



The screenshot shows the GitHub repository page for 'conflict-exercise'. The repository is public and has 2 branches and 0 tags. The main branch is selected. The README file is open, showing the title 'conflict-exercise' and a description: 'Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD (1C2025)'. It also mentions 'Esta es una línea nueva agregada al archivo en la rama main.' The commit history shows 4 commits by user 'arfernandez7691'.

conflict-exercise Public

main 2 Branches 0 Tags

Go to file Add file Code

arfernandez7691 Conflicto de merge resuelto 2163022 · 4 minutes ago 4 Commits

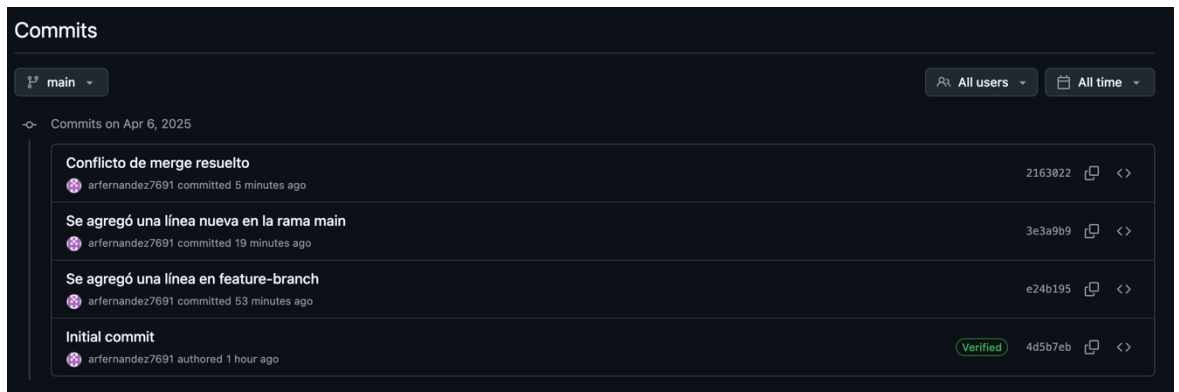
README.md Conflicto de merge resuelto 4 minutes ago

README

conflict-exercise

Este repositorio corresponde a la Actividad 3 de TP2 de la materia Programación 1 de la TUPaD (1C2025)

Esta es una línea nueva agregada al archivo en la rama main.



The screenshot shows the 'Commits' page for the 'conflict-exercise' repository. It lists the commit history for the main branch. The commits are ordered by time, with the most recent at the top. The commits are: 'Conflicto de merge resuelto' (2163022), 'Se agregó una línea nueva en la rama main' (3e3a9b9), 'Se agregó una línea en feature-branch' (e24b195), and 'Initial commit' (4d5b7eb). The 'Initial commit' is marked as 'Verified'.

Commits

main All users All time

Commits on Apr 6, 2025

Conflicto de merge resuelto 2163022

arfernandez7691 committed 5 minutes ago

Se agregó una línea nueva en la rama main 3e3a9b9

arfernandez7691 committed 19 minutes ago

Se agregó una línea en feature-branch e24b195

arfernandez7691 committed 53 minutes ago

Initial commit 4d5b7eb Verified

arfernandez7691 authored 1 hour ago