

TRABAJO PRÁCTICO N°2

PROGRAMACIÓN I

TUPAD - UTN

BELEN GROSSO

Ejercicio 1:

1. ¿Qué es GitHub?

GitHub es una plataforma o comunidad donde podemos compartir nuestros proyectos en forma de repositorios de manera pública o privada. Ofrece posibilidades de mostrar tu portafolio de trabajo a la hora de buscar empleo.

2. ¿Cómo crear un repositorio en GitHub?

Para crear un repositorio en GitHub lo que tenemos que hacer es crear una cuenta con usuario y contraseña. Luego en nuestro perfil debemos seleccionar el botón “+” en la parte superior derecha de la pantalla y clicar “nuevo repositorio”, donde colocaremos el nombre y decidiremos si este puede ser privado o público, agregar una descripción, entre otras configuraciones. Al terminar nos indicará los comandos a seguir para agregar este repositorio a Git.

3. ¿Cómo crear una rama en Git?

Para crear una rama en Git, debemos inicializar el repositorio creado con el comando “git init”, luego debemos escribir “git branch” seguido del nombre de la nueva rama que queremos crear.

4. ¿Cómo cambiar a una rama en Git?

Para cambiar a una rama debemos ingresar el comando “git checkout” seguido del nombre de la rama. Ejemplo: git checkout ejercicio.

5. ¿Cómo fusionar ramas en Git?

Para poder fusionar ramas debemos ingresar el comando “git merge” seguido de la rama que queremos fusionar.

6. ¿Cómo crear un commit en Git?

Para crear un commit primero debemos inicializar el repositorio, luego debemos escribir el comando “git add .” para que Git pueda rastrear este proyecto y por último el comando “git commit -m “ seguido de un mensaje que queremos dejar.

7. ¿Cómo enviar un commit a GitHub?

Para enviar un commit a GitHub nuestro repositorio debe estar anclado a Git.

8. ¿Qué es un repositorio remoto?

Un repositorio remoto es donde almacenamos dentro de la red nuestro proyecto para poder compartirlo al equipo con el que vamos a trabajar.

9. ¿Cómo agregar un repositorio remoto a Git?

Para crear un repositorio remoto en Git primero debemos ingresar el comando “git remote add origin” junto con la url donde está alojado nuestro repositorio.

10. ¿Cómo empujar cambios a un repositorio remoto?

Para empujar cambios a un repositorio remoto, debemos ingresar el comando “git push -u origin master” si es la primera vez que lo hacemos. De lo contrario, debemos escribir “git push”

11. ¿Cómo tirar de cambios de un repositorio remoto?

Para tirar de cambios se debe ingresar el comando “git pull”

12. ¿Qué es un fork de repositorio?

Un fork es una copia de otro repositorio creada en una cuenta distinta a la nuestra, donde se pueden efectuar cambios sin afectar a la original.

13. ¿Cómo crear un fork de un repositorio?

Para crear un fork de un repositorio primero debemos ingresar a GitHub e ir al repositorio que queremos copiar. Luego debemos seleccionar el botón “Fork” en la parte superior derecha de la pantalla.

14. ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

Primero debemos seleccionar el repositorio en el que queremos solicitar la extracción, luego debemos clicar la opción “Pull requests” en el menú de opciones. A partir de esto nos saldrá el home principal de las requests y debemos seleccionar el botón verde “New pull request”. Una vez realizado esto tenemos que ingresar las ramas a comparar, seguido de un título y descripción.

15. ¿Cómo aceptar una solicitud de extracción?

Para aceptar una solicitud, debemos primero revisar los cambios que se propusieron al proyecto que estamos construyendo y escribir una opinión al respecto. Al terminar, debemos aprobar la solicitud y enviarla.

16. ¿Qué es una etiqueta en Git?

Las etiquetas de Git son una porción del historial de Git que queremos destacar. Por ejemplo, confirmar la versión del proyecto.

17. ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta debemos ingresar el comando “git tag” seguido de la versión que queremos utilizar.

18. ¿Cómo enviar una etiqueta a GitHub?

Para enviar una etiqueta debemos utilizar el comando “git push -tags”.

19. ¿Qué es un historial de Git?

El historial es donde se registran todos los cambios realizados en el repositorio.

20. ¿Cómo ver el historial de Git?

Para ver el historial debemos escribir el comando “git reflog”.

21. ¿Cómo buscar en el historial de Git?

Para buscar en el historial podemos, por ejemplo, utilizar el comando “git reflog show”

- ¿Cómo borrar el historial de Git?

Podemos utilizar el comando “git reset --hard”.

22. ¿Qué es un repositorio privado en GitHub?

Un repositorio privado en GitHub es donde esta almacenado un trabajo forma que nadie pueda verlo.

23. ¿Cómo crear un repositorio privado en GitHub?

Cuando queremos crear nuestro repositorio, debemos clicar en la opción “Private” debajo de la descripción.

24. ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Para invitar a una persona a nuestro repositorio privado debemos primero enviar nuestra invitación haciendo click en “Settings” y luego seleccionar en “Collaborators”. Por último, ya podemos agregar a nuestros compañeros yendo al botón “Add people”.

25. ¿Qué es un repositorio público en GitHub?

Un repositorio publico es el lugar donde este alojado nuestro proyecto y puede ser visto por terceros.

26. ¿Cómo crear un repositorio público en GitHub?

Cuando queremos crear nuestro repositorio, debemos clicar en la opción “Public” debajo de la descripción.

27. ¿Cómo compartir un repositorio público en GitHub?

Para invitar a una persona a nuestro repositorio público debemos primero enviar nuestra invitación haciendo click en “Settings” y luego seleccionar en “Collaborators”. Por último, ya podemos agregar a nuestros compañeros yendo al botón “Add people”.

Ejercicio 2: <https://github.com/Belu1498/Ejercicio-2-Tp2-TUPaD>

```
$ git init
Initialized empty Git repository in C:/Users/belen/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2/.git/

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git add .

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git commit -m "Mi texto para Ejercicio 2"
[master (root-commit) f7e685e] Mi texto para Ejercicio 2
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Mi-texto.html

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git remote add origin https://github.com/Belu1498/Ejercicio-2-Tp2-TUPaD.git

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 227 bytes | 227.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Belu1498/Ejercicio-2-Tp2-TUPaD.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git branch
* master

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git branch nueva-rama

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git branch
* master
  nueva-rama

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git add .

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git commit -m "Se creo otro archivo .html de texto"
[master 07d98ea] Se creo otro archivo .html de texto
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 Mi texto-2.html

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (master)
$ git checkout nueva-rama
Switched to branch 'nueva-rama'

belen@Belu MINGW64 ~/OneDrive/Desktop/UTN Programacion I/Ejercicio-2-Tp2 (nueva-rama)
$ git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)
```

Ejercicio 3: <https://github.com/Belu1498/conflict-exercise-Ejercicio-3>

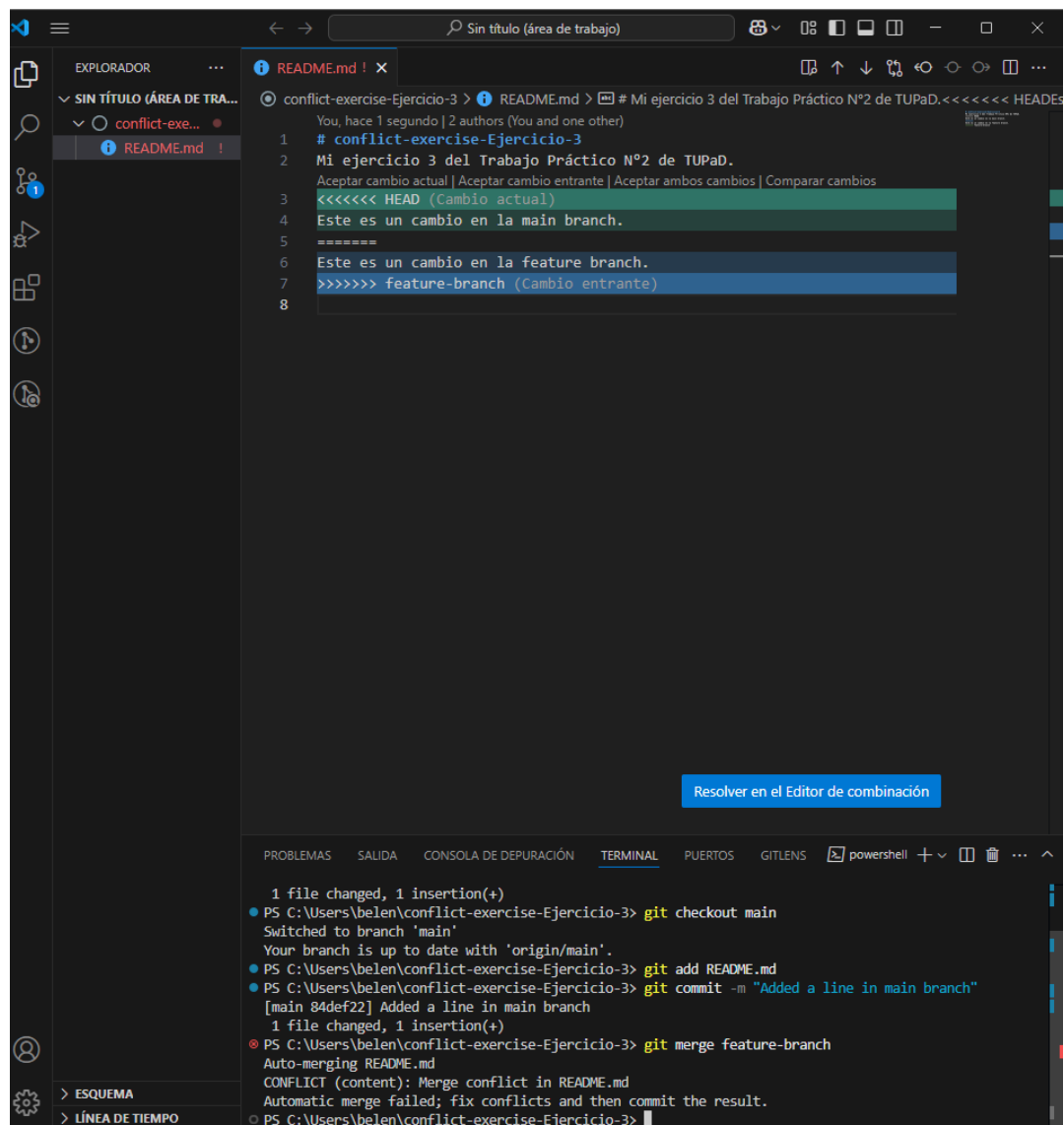
```
belen@Belu MINGW64 ~ (master)
$ git init
Reinitialized existing Git repository in C:/Users/belen/.git/

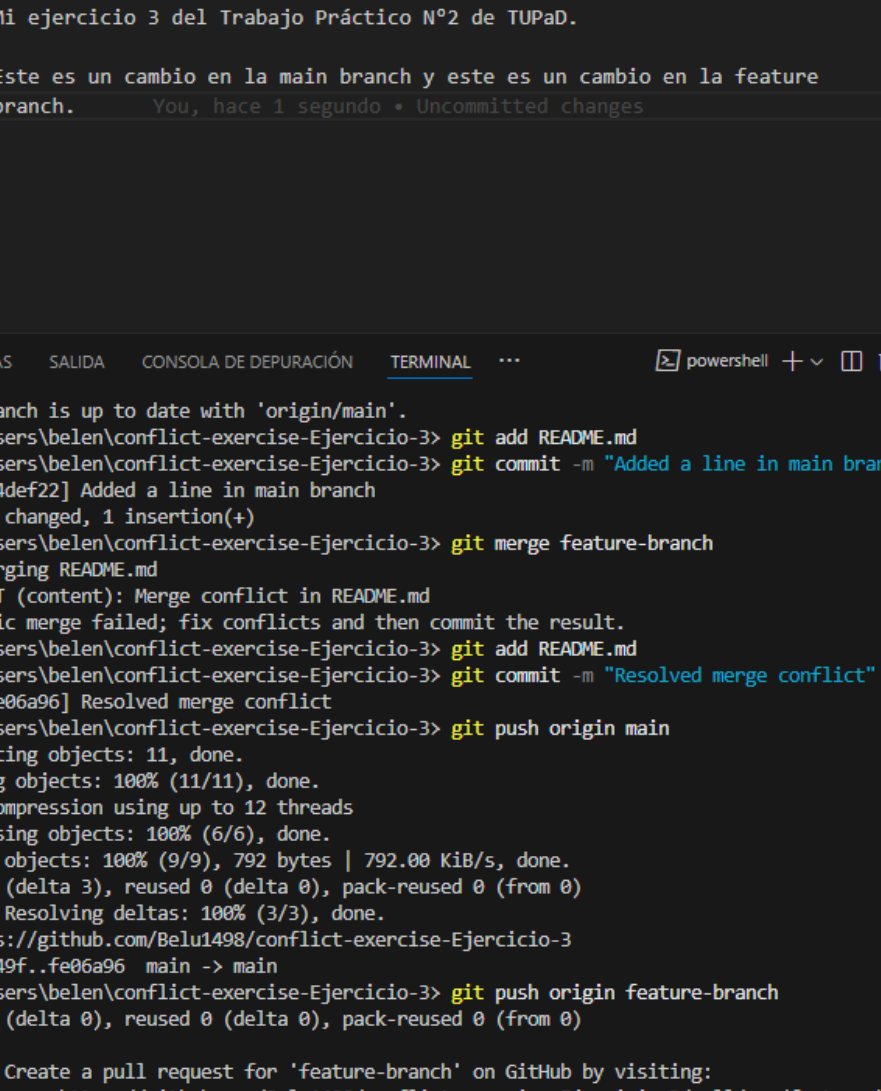
belen@Belu MINGW64 ~ (master)
$ git clone https://github.com/Belu1498/conflict-exercise-Ejercicio-3
Cloning into 'conflict-exercise-Ejercicio-3'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.

belen@Belu MINGW64 ~ (master)
$ cd conflict-exercise-Ejercicio-3

belen@Belu MINGW64 ~/conflict-exercise-Ejercicio-3 (main)
$ git checkout -b feature-branch
Switched to a new branch 'feature-branch'

belen@Belu MINGW64 ~/conflict-exercise-Ejercicio-3 (feature-branch)
$
```





The image shows a VS Code editor window with a file named `README.md` open. The file content is as follows:

```

You, hace 1 segundo | 2 authors (You and one other)
1 # conflict-exercise-Ejercicio-3
2 Mi ejercicio 3 del Trabajo Práctico N°2 de TUPaD.
3
4 Este es un cambio en la main branch y este es un cambio en la feature
  branch.
5

```

Below the editor, the **TERMINAL** tab is active, showing the following commands and output:

```

Your branch is up to date with 'origin/main'.
PS C:\Users\belen\conflict-exercise-Ejercicio-3> git add README.md
PS C:\Users\belen\conflict-exercise-Ejercicio-3> git commit -m "Added a line in main branch"
[main 84def22] Added a line in main branch
1 file changed, 1 insertion(+)
PS C:\Users\belen\conflict-exercise-Ejercicio-3> 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\belen\conflict-exercise-Ejercicio-3> git add README.md
PS C:\Users\belen\conflict-exercise-Ejercicio-3> git commit -m "Resolved merge conflict"
[main fe06a96] Resolved merge conflict
PS C:\Users\belen\conflict-exercise-Ejercicio-3> git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 12 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (9/9), 792 bytes | 792.00 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/Belu1498/conflict-exercise-Ejercicio-3
  20db49f..fe06a96  main -> main
PS C:\Users\belen\conflict-exercise-Ejercicio-3> 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/Belu1498/conflict-exercise-Ejercicio-3/pull/new/feature-branch
remote:
To https://github.com/Belu1498/conflict-exercise-Ejercicio-3
 * [new branch]   feature-branch -> feature-branch
PS C:\Users\belen\conflict-exercise-Ejercicio-3>

```