

TP 2 Programación

1)

- **¿Qué es GitHub?**

GitHub es una comunidad donde podemos compartir nuestros repositorios de forma pública o privada. Disponemos de un perfil similar a una red social, ya que podremos guardar nuestro código en la nube, trabajar en equipo y sobrescribir el trabajo de otros usuarios, hacer un seguimiento a los cambios de nuestro proyecto y colaborar con otras personas en proyectos de código abierto o privados.

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

Vamos a “nuevo repositorio” y le asignamos un nombre. Luego seleccionamos si queremos que sea público o privado. Finalmente seleccionamos la opción “crear repositorio.”

A partir de acá, debemos sincronizar nuestro repositorio local con nuestro repositorio remoto en GitHub:

...or create a new repository on the command line: si deseamos que nos guíe paso a paso

...or push an existing repository from the command line: si ya tenemos un repositorio local y deseamos mandarlo a GitHub. Para esto debemos escribir en la terminal:

```
$ git remote add origin https://github.com/tucanal/nombre-repo.git
```

```
$ git push -u origin master
```

- **¿Cómo crear una rama en Git?**

Para crear una nueva rama debemos utilizar el comando “git Branch”

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

Para cambiar de rama utilizamos el comando “git checkout”

- **¿Cómo fusionar ramas en Git?**

Para fusionar ramas utilizamos el comando “git merge”, situándonos con “git checkout” en la rama a la que deseamos añadirle los cambios. Seguido a esto, utilizamos “git merge” + la rama con los cambios. Por ejemplo: \$ git merge ramaCambios

- **¿Cómo crear un commit en Git?**

Una vez hechos los cambios en el archivo, procedemos a guardarlos en el commit de la siguiente manera:

`$git add .` para agregar todos los archivos o `$git add "nombre del archivo"` para hacerlo de manera singular.

Luego, utilizamos el comando `"git commit -m"` + el mensaje con las modificaciones.

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

En primer lugar, clonamos el repositorio de GitHub en nuestro repositorio local. Seguidamente hacemos las modificaciones deseadas en los archivos del repositorio y lo agregamos. Creamos un commit con los cambios y por último hacemos un push: `"git push origin rama con cambios"`

- **¿Qué es un repositorio remoto?**

Los repositorios remotos son versiones de un proyecto guardadas en la nube. Ayuda a trabajar colectivamente en proyectos grupales.

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

Utilizamos el comando `"git remote add + el nombre del repositorio remoto y la url"`

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

Utilizamos el comando `"git push origin "nombre_rama"`

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

Utilizamos el comando `"git pull nombre_rama"`

- **¿Qué es un fork de repositorio?**

Un Fork es una copia de un repositorio que creamos en nuestra cuenta con la finalidad de poder editarlo sin modificar el original.

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

Abrimos GitHub y buscamos el repositorio a copiar. Sobre el margen derecho hacemos click en `"Fork"` y seguidamente seleccionamos la cuenta donde queremos crear el fork.

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

Vamos a la solapa “pull requests” y clickeamos donde dice “new pull request”. Luego hacemos click donde dice “créate pull request” y luego mencionamos el motivo de cambio realizado y sus ventajas.

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

En la sección de “pull requests” el autor del repositorio podrá ver el mensaje que le enviaron y considerar hacer efectivo el cambio.

- **¿Qué es un etiqueta en Git?**

Sirve para destacar puntos importantes del historial como nuevas versiones del repositorio.

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

Para crear una etiqueta ligera: `git tag nombre_etiqueta`

Para crear una etiqueta anotada: `git tag -a nombre_etiqueta -m "Descripción de la versión"`

Para subir una etiqueta: `git push origin nombre_etiqueta`

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

Luego de crear la etiqueta en el repositorio local utilizamos el siguiente comando para pushear:

`git push origin v1.0`

o: `git push origin -tags` para empujar todas las etiquetas creadas

- **¿Qué es un historial de Git?**

Es un orden de todos los cambios realizados en un repositorio de Git.

- **¿Cómo ver el historial de Git?**

Para esto utilizamos el comando: `Git log`

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

Si recordamos una palabra clave podemos usar `-grep`: `git log --grep="palabra clave"`

Si buscamos cambios en un archivo específicos podemos usar: `git log --nombre_del_archivo`

Para buscar commits en un rango de fechas podemos usar `--since` y `--until`:

`git log --since="2024-01-01" --until="2024-01-31"`

También podemos buscar por autor: `git log --author "Nombre del Autor"`

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

git reset: elimina del stage todos los archivos y carpetas del proyecto.

git reset nombreArchivo elimina del stage el archivo seleccionado.

git reset nombreCarpeta: elimina del stage todos los archivos de esa carpeta.

git reset nombreArchivo elimina el archivo del stage.

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

Es un repositorio en el que sólo tienen acceso al contenido, usuarios que han sido autorizados por el autor.

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

Vamos al margen superior derecho y seleccionamos “+” y luego “new repository”. Completamos la información requerida y en la sección de privacidad tildamos sobre “private”

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

Nos dirigimos a “settings” y luego seleccionamos “Collaborators”. Luego clickeamos sobre “add people” e ingresamos el nombre del usuario que deseamos invitar.

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

Un repositorio público es aquel en el que todo usuario de GitHub tiene acceso sin un filtro previo. De esta manera podrá ver el repositorio, clonarlo y hasta ayudar en el proyecto de manera colaborativa.

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

Al momento de crear el repositorio, tildamos la opción de “public”

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

Podemos compartir un repositorio público compartiendo en link del mismo. También podemos entrar a nuestro repositorio y copiar la URL (<> Code)

2) Realizar la siguiente actividad:


- Crear un repositorio.
 - Dale un nombre al repositorio.
 - Elije 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 ().*


Owner *

 equimdq ▾

 /


Repository name *

TP-2Programacion


 The repository TP-2Programacion already exists on this account.

Great repository names are short and memorable. Need inspiration? How about **bookish-guacamole** ?

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.](#)

 You are creating a public repository in your personal account.

Create repository

- Agregando un Archivo

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

- o Realiza los comandos `git add .` y `git commit -m "Agregando mi-archivo.txt"`

en la línea de comandos.

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

```
PS C:\Users\Eze\programacion1\TP2-Programacion> git add .
PS C:\Users\Eze\programacion1\TP2-Programacion> git commit -m "Agregando mi-archivo.txt"
[main (root-commit) 410bacd] Agregando mi-archivo.txt
 1 file changed, 1 insertion(+)
 create mode 100644 mi_archivo.txt
PS C:\Users\Eze\programacion1\TP2-Programacion> git push origin master
error: src refspec master does not match any
error: failed to push some refs to 'https://github.com/equimdq/TP-2Programacion.git'
PS C:\Users\Eze\programacion1\TP2-Programacion> git push origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 244 bytes | 244.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'main' on GitHub by visiting:
remote:   https://github.com/equimdq/TP-2Programacion/pull/new/main
remote:
To https://github.com/equimdq/TP-2Programacion.git
 * [new branch]      main -> main
PS C:\Users\Eze\programacion1\TP2-Programacion> 
```

The screenshot shows the GitHub web interface for the repository 'equimdq/TP-2Programacion'. At the top, a notification bar indicates 'main had recent pushes 5 minutes ago' with a 'Compare & pull request' button. Below this, the branch 'main' is selected, showing '2 Branches' and '0 Tags'. A search bar and buttons for 'Add file' and 'Code' are visible. A status bar indicates 'This branch is 1 commit ahead of, 2 commits behind master'. The commit history shows a single commit by 'equimdq' titled 'Agregando mi-archivo.txt' with hash '410bacd' from 6 minutes ago. The file 'mi_archivo.txt' is listed as the only change in this commit.

- Creando Branchs

- o Crear una Branch

- o Realizar cambios o agregar un archivo

- o Subir la Branch

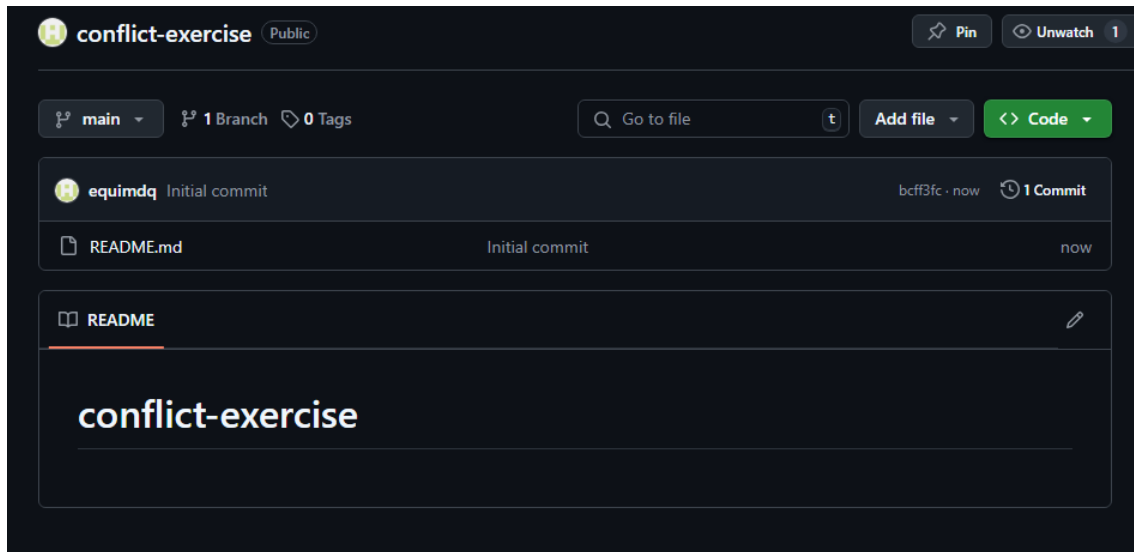
```
PS C:\Users\Eze\programacion1\TP2-Programacion> git checkout -b rama_1
Switched to a new branch 'rama_1'
PS C:\Users\Eze\programacion1\TP2-Programacion> echo "Este es un archivo nuevo en la rama rama_1" > nuevo_archivo.txt
PS C:\Users\Eze\programacion1\TP2-Programacion> git add .
PS C:\Users\Eze\programacion1\TP2-Programacion> git commit -m "Agregué el archivo: nuevo_archivo.txt"
[rama_1 3e85c39] Agregué el archivo: nuevo_archivo.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 nuevo_archivo.txt
PS C:\Users\Eze\programacion1\TP2-Programacion> git push origin rama_1
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 359 bytes | 359.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'rama_1' on GitHub by visiting:
remote:   https://github.com/equimdq/TP-2Programacion/pull/new/rama_1
remote:
To https://github.com/equimdq/TP-2Programacion.git
 * [new branch]      rama_1 -> rama_1
```

Default					
Branch	Updated	Check status	Behind / Ahead	Pull request	
<code>master</code>	39 minutes ago		<div>Default</div>		

Your branches					
Branch	Updated	Check status	Behind / Ahead	Pull request	
<code>rama_1</code>	4 minutes ago		<div>2 2</div>		
<code>main</code>	26 minutes ago		<div>2 1</div>		

Active branches					
Branch	Updated	Check status	Behind / Ahead	Pull request	
<code>rama_1</code>	4 minutes ago		<div>2 2</div>		
<code>main</code>	26 minutes ago		<div>2 1</div>		

Paso 1: Crear un repositorio en GitHub



Paso 2: Clonar el repositorio a tu máquina local

```
Símbolo del sistema
Microsoft Windows [Versión 10.0.19045.5608]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Eze>git clone https://github.com/tuusuario/conflict-exercise.git
Cloning into 'conflict-exercise'...
remote: Repository not found.
fatal: repository 'https://github.com/tuusuario/conflict-exercise.git/' not found

C:\Users\Eze>git clone https://github.com/equimdq/conflict-exercise.git
Cloning into 'conflict-exercise'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.

C:\Users\Eze>
```

Paso 3: Crear una nueva rama y editar un archivo

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

```
PS C:\Users\Eze\programacion1\TP2-Programacion> code README.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git add README.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git commit -m "Se agrego una linea en feature-branch"
[rama_1 86471ba] Se agrego una linea en feature-branch
1 file changed, 1 insertion(+)
create mode 100644 README.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\Eze\programacion1\TP2-Programacion> code readme.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git add readme.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git commit -m "Added a line in the main branch"
[main 580e185] Added a line in the main branch
1 file changed, 1 insertion(+)
create mode 100644 readme.md
```


Paso 5: Hacer un merge y generar un conflicto

```
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
1 <<<<<< HEAD (Current Change)
2 Este es un cambio en la main branch.
3 =====
4 Este es un cambio en la rama correspondiente a feature-branch.
5 >>>>>> feature-branch (Incoming Change)
6
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

```
PS C:\Users\Eze\programacion1\TP2-Programacion> git checkout main
Already on 'main'
Your branch is ahead of 'origin/main' by 2 commits.
(use "git push" to publish your local commits)
PS C:\Users\Eze\programacion1\TP2-Programacion> code readme.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git add readme.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git commit -m "cambio en main branch"
[main 953c29a] cambio en main branch
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\Eze\programacion1\TP2-Programacion> git checkout feature-branch
Switched to branch 'feature-branch'
PS C:\Users\Eze\programacion1\TP2-Programacion> git add readme.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git commit -m "cambio en feature branch"
[feature-branch 0441b74] cambio en feature branch
 1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\Users\Eze\programacion1\TP2-Programacion> git checkout main
Switched to branch 'main'
Your branch is ahead of 'origin/main' by 3 commits.
(use "git push" to publish your local commits)
PS C:\Users\Eze\programacion1\TP2-Programacion> 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.
```

Paso 6: Resolver el conflicto

```
i readme.md X
C: > Users > Eze > Programacion1 > TP2-Programacion > i readme.md
1 Este es un cambio en la main branch y en la rama correspondiente a feature-branch
```

```

PS C:\Users\Eze\programacion1\TP2-Programacion> git add readme.md
PS C:\Users\Eze\programacion1\TP2-Programacion> git status
On branch main
Your branch is ahead of 'origin/main' by 3 commits.
  (use "git push" to publish your local commits)

All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
  modified:   readme.md

PS C:\Users\Eze\programacion1\TP2-Programacion> git commit -m "Conflicto de merge resuelto"
[main fa5767c] Conflicto de merge resuelto
PS C:\Users\Eze\programacion1\TP2-Programacion>

```

Paso 7: Subir los cambios a GitHub

```

PS C:\Users\Eze\programacion1\TP2-Programacion> git push origin main
Enumerating objects: 16, done.
Counting objects: 100% (16/16), done.
Delta compression using up to 12 threads
Compressing objects: 100% (13/13), done.
Writing objects: 100% (15/15), 1.50 KiB | 769.00 KiB/s, done.
Total 15 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/equimdq/TP-2Programacion.git
  410bacd..fa5767c  main -> main

```


Paso 8:

Los cambios se hicieron en el repositorio local pero no logre que se vean reflejados en GitHub a pesar de forzar el push



```



PS C:\Users\Eze\Programacion1\TP2-Programacion> git push origin new-branch
Everything up-to-date


```


 **conflict-exercise** Public


[Pin](#) [Unwatch](#) 1


 **bcff3fc** 


 **1** Branch  **0** Tags



[Code](#) 


 **equimdq** Initial commit

bcff3fc · 1 hour ago  **1** Commit

 **README.md**

Initial commit

1 hour ago

 **README**

conflict-exercise