

**Java y git, codigos e instalación.**

Ronald Felipe Guayambuco Muñoz

Universidad de Cundinamarca Extensión Chía

Programación II

William Alexander Matallana Porras

20 de febrero de 2025

Introducción .....	3
Objetivo.....	4
Instalación de Java e IntelliJ.....	5
Creación de un proyecto y subida a repositorio .....	8
Git Pull y Git Revert.....	14
Conclusión .....	15
Referencias .....	16

## **Introducción**

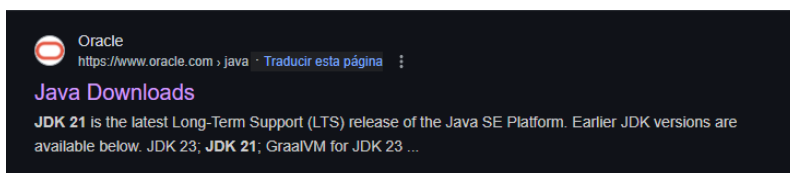
Mediante este documento se van a explicar el cómo se realiza la instalación de git, IntelliJ, Java y como usar estas mismas junto a la página de repositorios GitHub, en suma, a esto se harán explicaciones de ciertos códigos los cuales son de utilidad a la hora de crear un repositorio y eventualmente subirlo a la misma plataforma GitHub

## **Objetivo**

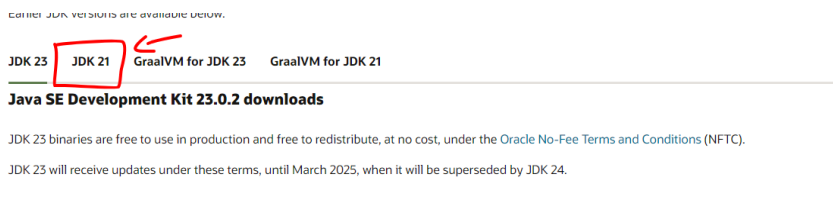
El objetivo de este documento es realiza una guía paso a paso de como instalar jdk21, su enlace con GitHub y como crear un repositorio junto a la explicación de las líneas usadas durante este método, junto a esto de adjunta una breve explicación de lo que es un comando git revert y cómo funciona, de igual manera este documento tendrá como otro objetivo servir de guía a la hora de exportar e importar un repositorio a la plataforma GitHub

## Instalación de Java e IntelliJ

Para realizar la instalación de java tenemos que obtener lo que sería el JDK21, este si entramos a nuestro navegador de preferencia y en la barra de búsqueda ingresamos “JDK21” lograremos encontrar un enlace a la página de Oracle.

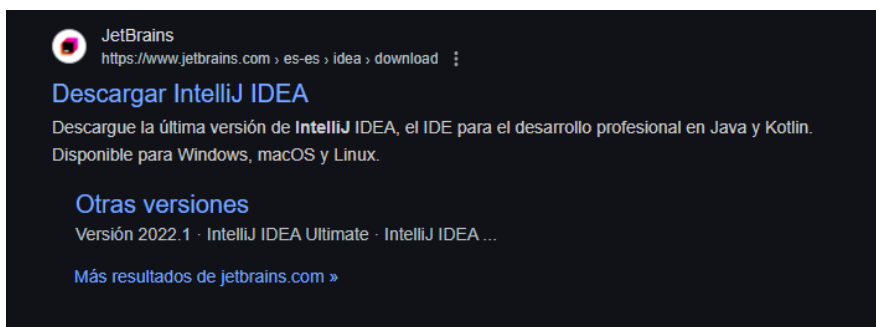


Ingresamos a él enlace presentado y tenemos que seleccionar en la página de java la opción llamada JDK21



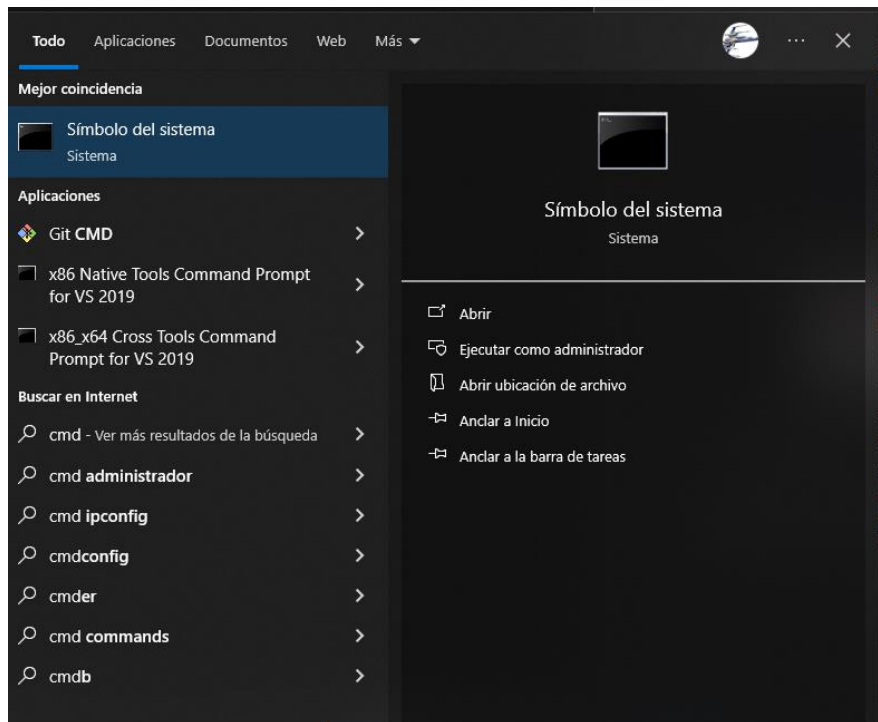
Después seleccionamos la opción windows y seleccionamos la opción acorde a los bits manejados en la computadora (por defecto la mayoría son de x64 bits).

Una vez hallamos descargado JDK21 en nuestra computadora podemos empezar la descarga de Git, en este caso, volvemos al navegador y buscamos “intellij” y entramos al link proporcionado por JetBrains.



Procedemos a descargar de su página el ejecutable .exe y de igual forma que como instalamos JDK21, continuamos los pasos de instalación.

Luego de descargar el instalador de JDK21 y Git, si queremos revisar cual versión de JDK/Java estamos manejando actualmente en nuestra computadora, se tiene que buscar el CMD de la computadora, podemos realizarlo buscando “cmd” en la barra de búsqueda la computadora.



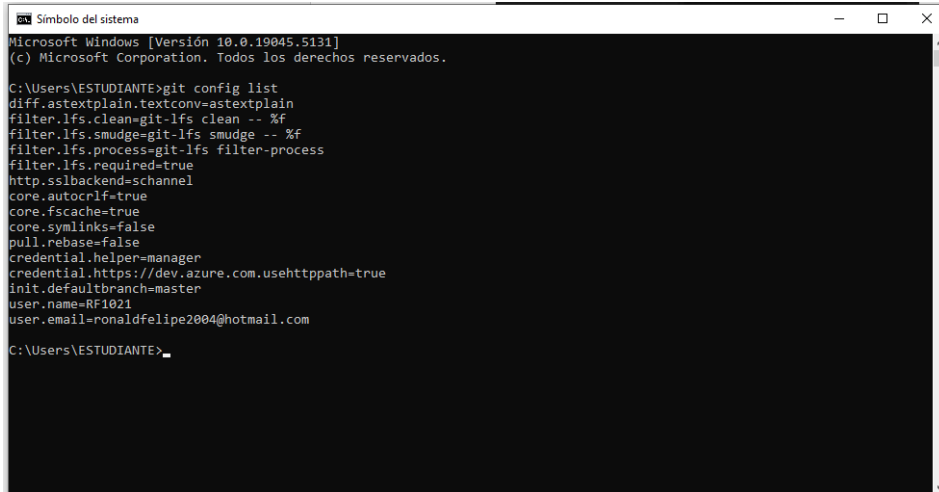
Una vez abrir el cmd de sistema podemos escribir el siguiente comando,

- git --version
- Java --version

Para la configuración de nuestras cuentas y usuarios en la computadora debemos de ingresar los siguientes comandos

- Git config --global user.name “nombre de usuario”
- Git config --global user.email “correo electrónico”

Debemos de ingresar el código Git config list y obtendremos la siguiente lista



```

Microsoft Windows [Versión 10.0.19045.5131]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\ESTUDIANTE>git config list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=schannel
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=RF1021
user.email=ronaldfelipe2004@hotmail.com

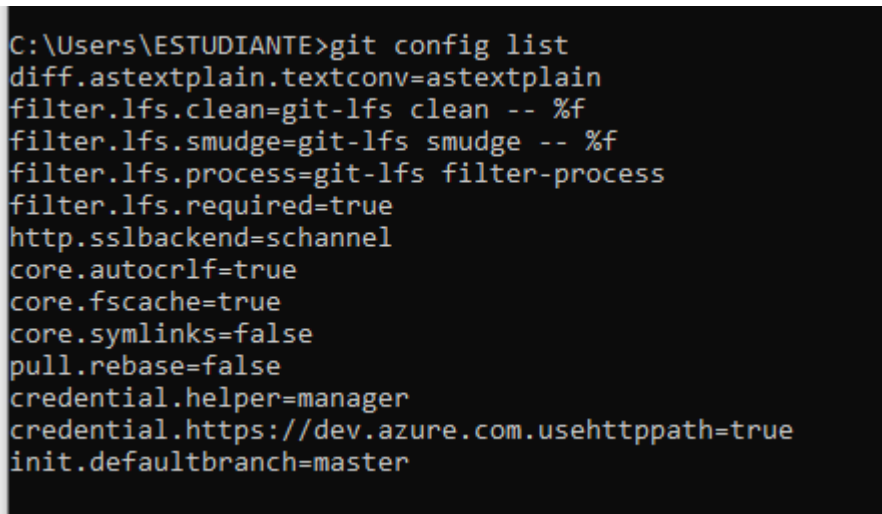
C:\Users\ESTUDIANTE>

```

Para remover las cuentas configuradas mediante este comando podemos ingresar.

- Git config - - global - -unset user.email
- Git config - - global - -unset user.name

Volvemos a realizar un git config list y veremos un cambio en el user y email y se visualizará que el email y nombre ingresados ya no están.



```

C:\Users\ESTUDIANTE>git config list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=schannel
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master

```

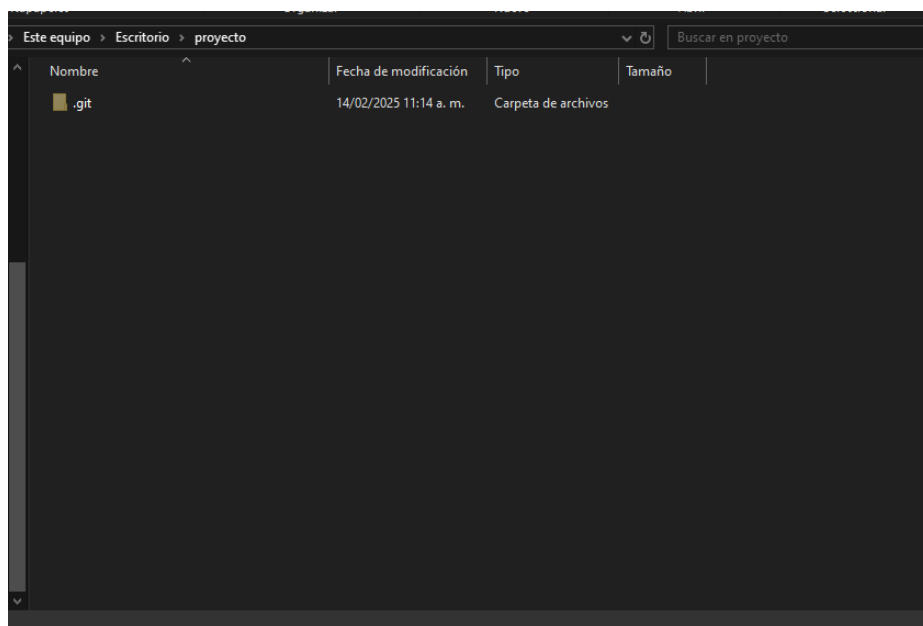
De esta manera podremos revisar si nuestra versión de Java instalada es la correcta, junto a la versión instalada de Git, y configurar nuestro usuario de Git en la computadora.

## Creación de un proyecto y subida a repositorio

Una vez realizado todo esto podremos ejecutar intelliJ, abrimos un proyecto nuevo y ahora lo que se realizara es la configuración de intelliJ y su vinculación a GitHub, hay que tener en cuenta que para realizar estos siguientes pasos debemos de poseer una cuenta de GitHub, de lo contrario no habrá manera de combinar la cuenta con la aplicación.

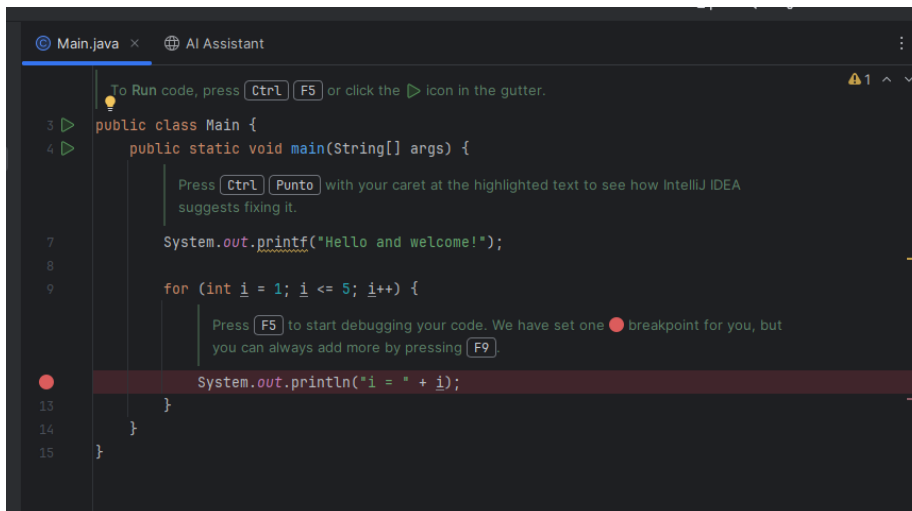
Una vez estemos en la pestaña del programa seleccionamos las 3 líneas y vamos a la opción setting y una vez estemos dentro encontraremos una barra de búsqueda, ahí ingresamos la palabra “Git”, una vez veamos las opciones encontraremos una pestaña llamada GitHub y la seleccionamos, tendremos un pequeño menú y le damos a un icono de “+” ahí podremos hacer un logeo de GitHub y unirlo a nuestra computadora.

Una vez realizada la configuración e instalación de intelliJ , java, y su enlace con GitHub, debemos de ir a nuestro escritorio y crear una carpeta, insertar su respectivo nombre, una vez hecho esto debemos de dar click derecho y seleccionar la opción open git bash, ingresamos el comando “git init” para establecer nuestro repositorio local, ahora tendremos una carpeta con nuestro proyecto. (debemos de ingresar un git add . Para añadir todos los archivos, y luego un git commit -m “mensaje”)



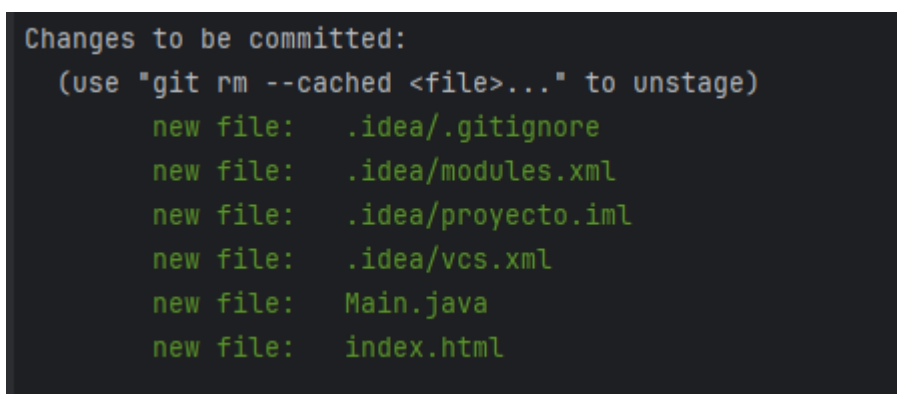
Ahora vamos a hacer un código y subirlo a un repositorio de GitHub, en este caso se usará el ejercicio de ejemplo entregado por el mismo intelliJ para este procedimiento





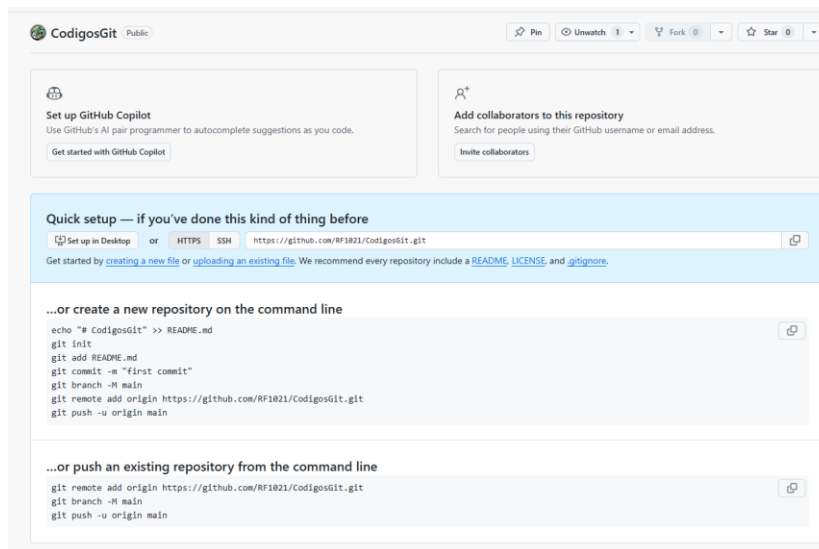
Una vez tenemos nuestro código podemos proceder a la apertura de una terminal en el archivo y una vez la tengamos debemos de realizar un “git status”

Con este código veremos en la consola como el sistema detecta los archivos visibles en ella



Ahora realizaremos un “git commit -m” con este código podremos añadir un commit /mensaje en el repositorio en este caso se escribirá “guía”

Ahora pasaremos a la creación del repositorio, en GitHub debemos ir a crear repositorio y obtendremos la siguiente ventana



Tomamos la serie de comandos que GitHub nos entrega para crear el repositorio en este caso:

```
git remote add origin https://github.com/RF1021/CodigosGit.git
```

```
git branch -M main
```

```
git push -u origin main
```

Una vez puestos los comandos tendremos ya un repositorio en GitHub !

```
PS C:\Users\user\Desktop\proyecto> git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 12 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 1.23 KiB | 1.23 MiB/s, done.
Total 9 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/RF1021/mi-proyecto.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
PS C:\Users\user\Desktop\proyecto>
```

RF1021 guia		54d4c82 · 14 minutes ago	1 Commit
.idea	guia	14 minutes ago	
Main.java	guia	14 minutes ago	
index.html	guia	14 minutes ago	

Ahora vamos a ver cómo se realiza un git clone, el git clone sirve para una vez tengamos que realizar un clonado de un repositorio de GitHub, podremos almacenarlo localmente en nuestra computadora, para esto tenemos que volver a nuestra carpeta, o crear una nueva, y hacer un git bash

Una vez abramos nuestra ventana con git bash debemos de ingresar la línea “git clone” y poner el link de GitHub que tengamos.

Una vez con todo esto veremos lo que son las ramas o branches, usaremos en este caso nuestra carpeta que realizamos para mostrar los códigos. Una vez estamos en nuestro archivo debemos de ingresar el comando “git branch”, con este comando lograremos visualizar todas las ramas que tengamos presentes

```
PS C:\Users\user\Desktop\proyecto> git branch
* main
  master
```

Ahora ingresaremos el comando “git switch -c (nombre)” con esta línea lo que haremos es crear una nueva rama y de igual manera movernos a esta, en este caso crearemos una branch llamada “ejemplo”

```
PS C:\Users\user\Desktop\proyecto> git branch
* ejemplo
  main
  master
```

Ahora lo que haremos es volver a la branch “main” para esto utilizaremos el comando “git switch main”

```
PS C:\Users\user\Desktop\proyecto> git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Y ahora subiremos nuestra nueva rama a GitHub , para esto debemos de realizar un git status, y luego un “git push origin ejemplo”

```
PS C:\Users\user\Desktop\proyecto> git push origin ejemplo
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'ejemplo' on GitHub by visiting:
remote:   https://github.com/RF1021/mi-proyecto/pull/new/ejemplo
remote:
To https://github.com/RF1021/mi-proyecto.git
 * [new branch]      ejemplo -> ejemplo
PS C:\Users\user\Desktop\proyecto>
```

Si de esta forma deseamos eliminar una branch solo debemos de escribir “git branch -D (nombre)”

Con esto podemos empezar el uso de el git fetch, el git Fetch nos servirá para obtener las actuaciones del repositorio remoto

Ahora veremos git branch -r, este comando nos servira para ver las branch que se presentan actualmente en nuestro repositorio, para este ejemplo se crearan 2 nuevos branches para eso usamos un git switch -c “ejemploA” y “ejemploB”, luego hacemos un git push origin para ingresar las nuevas branch al repositorio y deberíamos tener el siguiente resultado:



Ahora realizaremos un git branch -r

Y ahora veremos las ramas remotas

```
PS C:\Users\ESTUDIANTE\Desktop\Codigo Git\Codigos git> git branch -r
origin/HEAD -> origin/main
origin/ejemploA
origin/ejemploB
origin/main
```

Ahora veremos el comando git log, git reflog y git log --oneline

Si ingresamos git log podremos ver este resultado

```
PS C:\Users\ESTUDIANTE\Desktop\Codigo Git\Codigos git> git log
commit 8c70a8a99c1b04510c4c7776578e00c0b1a9df9 (HEAD -> ejemploB, origin/main, origin/ejemploB, origin/ejemploA, origin/HEAD, main, ejemploA)
Author: RF1021 <ronaldfeipe2004@hotmail.com>
Date: Thu Feb 20 17:36:39 2025 -0500
```

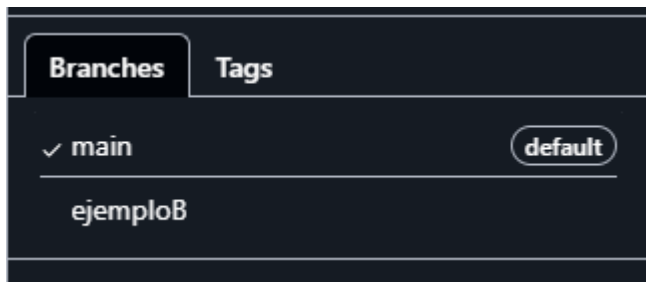
Ahora con git reflog podremos ver los cambios en el repositorio

Y con git log --oneline obtendremos el historial de logs de una forma menos detallada

Como siguiente se explica el git merge+ (--rebase) lo que hace este codigo , es que, en lugar de crear un commit de merge , a cambio toma todos los commit de la rama actual y los mete a un “stash”, lo que hace es el evitar los merge commit y mantiene un historial ordenado

El git push origin --delete rama funciona para realizar una eliminación de manera remota de git, en este caso tomaremos el repositorio que creamos y vamos a eliminar una de las ramas, en este caso la rama “ejemploA”,

```
PS C:\Users\user\Desktop\Codigos git\CodigosGit> git push origin --delete ejemploA
To https://github.com/RF1021/CodigosGit.git
- [deleted]      ejemploA
```



Como resultado en la página de GitHub se ve como se removió el branch

## Git Pull y Git Revert

la función del git pull y el git revert / devolver commit, El propósito del git pull será la importación de cambios a nuestro dispositivo local

En cuanto al comando git revert, lo que realizamos con este comando es el eliminar o devolver un commit, lo que realiza en sí es crear un nuevo commit que revierte los cambios al anterior, se puede usar este comando una vez se hayan subido cambios a GitHub los cuales se deseen revertir

Si tomamos por ejemplo el siguiente historial

a1b2c3d Agregando nueva función

4e5f6g7 Arreglando bug en el código

8h9i0j1 Primer commit

Si realizamos un git revert a cualquiera de estos 3 commits simplemente deberíamos poner “git revert 4e5f6g7”

## **Conclusión**

Durante el trayecto de la creación del documento se ha evidenciado e aprendido como de debe de crear un repositorio, como subirlo a un GitHub como repositorio, como hacer un pull y traer archivos a nuestro dispositivo local, el cómo enlazar nuestra cuenta a intelliJ, y importantemente la descarga e instalación de las herramientas utilizadas durante el proceso.

## Referencias

Chat Gpt , porcentaje de uso: 8% (explicación del git revert)