

SENA

JUAN CAMILO SARRAZOLA

Curso: Git y GitHub

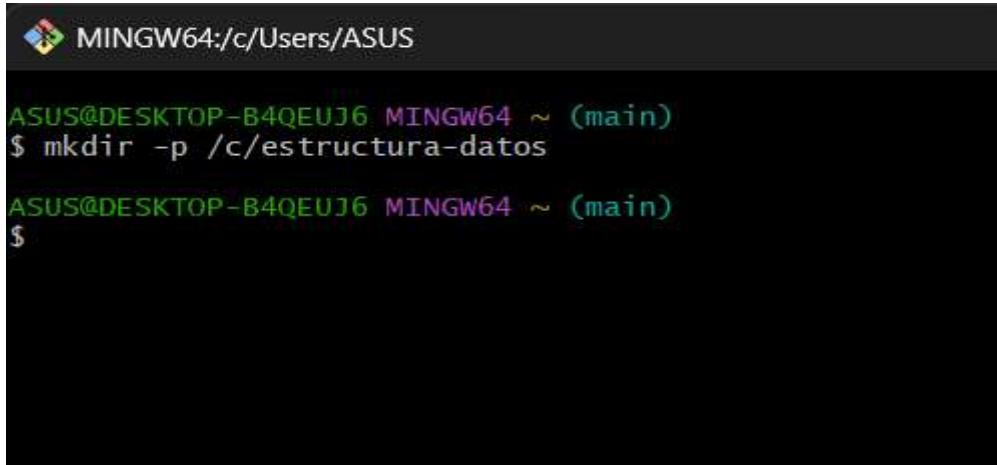
Profesor: Nolfi

Ficha: 3169901

2025

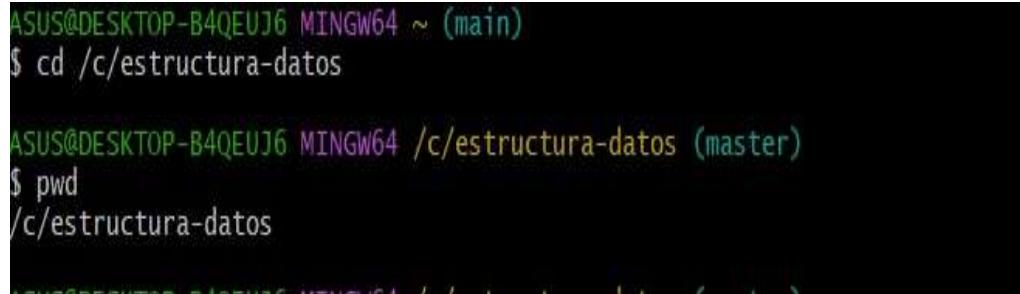
Desarrollo del tema:

1. Cree la carpeta de su proyecto (ej: C:\estructura-datos).



```
MINGW64:/c/Users/ASUS
ASUS@DESKTOP-B4QEIJ6 MINGW64 ~ (main)
$ mkdir -p /c/estructura-datos
ASUS@DESKTOP-B4QEIJ6 MINGW64 ~ (main)
$
```

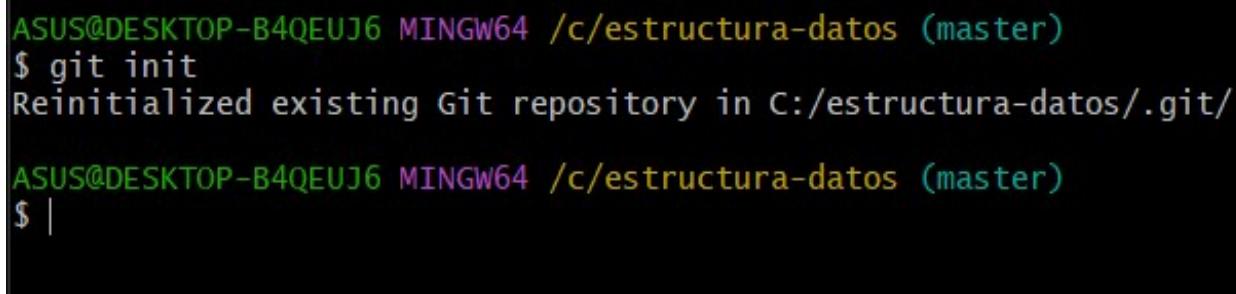
2. Navegue a la carpeta en su terminal.



```
ASUS@DESKTOP-B4QEIJ6 MINGW64 ~ (main)
$ cd /c/estructura-datos

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ pwd
/c/estructura-datos
```

3. Ejecute \$ git init.



```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git init
Reinitialized existing Git repository in C:/estructura-datos/.git/

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ |
```

Ejercicios prácticos:

Ejercicio 2: Ciclo Básico de Commit (Asumiendo que ya ejecutó git init en una carpeta llamada estructura-datos):

1. Cree dos archivos nuevos: Programa_01.cpp y notas.txt (simule un archivo Untracked).

```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ echo "// Programa 01 - Ejercicio inicial" > Programa_01.cpp

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ echo "Notas iniciales" > notas.txt

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$
```

2. Verifique su estado: \$ git status.

```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Programa_01.cpp
    notas.txt

nothing added to commit but untracked files present (use "git add" to track)

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ |
```

3. Agregue solo Programa_01.cpp al Staging Area: \$ git add Programa_01.cpp.

```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git add programa_01.cpp

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    Programa_01.cpp
    notas.txt

nothing added to commit but untracked files present (use "git add" to track)

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ |
```

4. Confirme el cambio de Programa_01.cpp: \$ git commit -m "feat: Primer ejercicio resuelto".

```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git commit -m "feat: Primer ejercicio resuelto"
[master (root-commit) e7033c3] feat: Primer ejercicio resuelto
 1 file changed, 1 insertion(+)
 create mode 100644 Programa_01.cpp

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ |
```

5. Modifique el archivo Programa_01.cpp (añada una línea de código) y modifique notas.txt

```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ echo "// Línea nueva de código" >> Programa_01.cpp

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ echo "Nueva nota agregada" >> notas.txt

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   Programa_01.cpp

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    notas.txt

no changes added to commit (use "git add" and/or "git commit -a")

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ |
```

6. Ejecute git add . para añadir ambos al Staging Area.

```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git add .
warning: in the working copy of 'Programa_01.cpp', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'notas.txt', LF will be replaced by CRLF the next time Git touches it

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   Programa_01.cpp
    modified:   notas.txt

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ |
```

7. Confirme ambos cambios: \$ git commit -m "fix: Corregir lógica de Programa_01 y añadir notas"

```
ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$ git commit -m "fix: Corregir lógica de Programa_01 y añadir notas"
[master eca7b86] fix: Corregir lógica de Programa_01 y añadir notas
 2 files changed, 3 insertions(+)
 create mode 100644 notas.txt

ASUS@DESKTOP-B4QEIJ6 MINGW64 /c/estructura-datos (master)
$
```

1. **Defina qué es el Staging Area y cuál es su propósito en el ciclo de trabajo de Git.**

El Staging Area (área de preparación) es donde se guardan temporalmente los archivos que se van a confirmar.

Permite revisar y seleccionar qué cambios se incluirán en el próximo commit.

2. **Si un archivo ha sido modificado en el directorio de trabajo, pero nunca se ha ejecutado git add sobre él, ¿en qué estado se encuentra?**

Está en estado Modified (modificado, pero no preparado).

3. **¿Qué ocurre internamente en Git cuando se ejecuta git commit?**

Git toma una instantánea de los archivos del Staging Area y la guarda en la base de datos interna .git como un nuevo commit con su mensaje, autor y fecha.

4. **¿Qué comando simplificado puede usar para mover archivos Tracked y Modified directamente al commit, saltando el git add?**

El comando simplificado que se usa para mover archivos “Tracked” (rastreables) y “Modified” (modificados) directamente al commit, saltándose el paso del git add, es:

```
git commit -a -m "Mensaje del commit"
```