Segunda Consigna

- git clone <https://nombre-del-repositorio> : Este comando sirve para descargar código abierto existente desde un repositorio remoto (GitHub). Básicamente, crea una copia idéntica a la última versión del proyecto en un repositorio y lo guarda en nuestra computadora.

- git init : Crea un nuevo repositorio de Git. Puede utilizarse para convertir un proyecto existente y sin versión en un repositorio de Git, o para inicializar un nuevo repositorio vacío.

- git status: Este comando entreva toda la información necesario de la Rama actual. Nos dice si la rama está actualizada a la fecha; si acaso hay algún archivo por commitear, pushear o pullear; si hay archivos añadidos, sin añadir o untracked; y si hay archivos creados, modificados o eliminados.

- Git add <*archivo o .*> : Cuando creamos, modificamos o eliminamos un archivo, estos cambios sucederán en nuestro repositorio local y no serán incluidos en el siguiente commit. Para ello usamos el comando git add, que incluye los cambios de el o los archivos para que estén preparados para el siguiente commit.

- Git commit -m "Mensaje del Commit": Una vez que llegamos a cierto punto en el desarrollo, queremos guardar los cambios (tal vez luego de una tarea específica o conflicto). Git commit nos permite hacer un checkpoint en el proceso del desarrollo al cual podemos volver luego si es requerido.

- Git log: Muestra los registros de los commit. Enumera las confirmaciones a las que se puede llegar mostrando números de serie predeterminados y únicos.

- Git push origin <nombre de la rama>: Luego de guardar nuestros cambios (commit), la siguiente cosa que se quiere hacer es enviar los cambios al servidor remoto, en este caso GitHub. Git push sube los commits al repositorio remoto. Importante: Git push solo sube los cambios que han sido confirmados.

- Git pull <remote>: Este comando es usado para obtener las actualizaciones desde el repositorio remoto. Es una combinación entre *git fetch* y *git merge*, lo cual significa que, cuando usamos git pull, se obtienen los cambios del repositorio remoto (git fetch) e inmediatamente aplica los últimos cambios al repositorio local (git merge).

- Git branch <nombre de la rama>: Las ramas son sumamente importantes en el mundo de git. Usando ramas, varios desarrolladores son capaces de trabajar en paralelo simultáneamente en el mismo proyecto. Podemos usar el comando git branch para crear una rama.

- Git branch --list: Sirve para listar las ramas actuales.

- Git branch -d <nombre de la rama>: Sirve para eliminar una rama.

- Git checkout <nombre de la rama>: Para trabajar en una rama, primero debemos cambiarnos a ella. Usando git checkout podemos hacer este cambio saltando de una rama a otra.