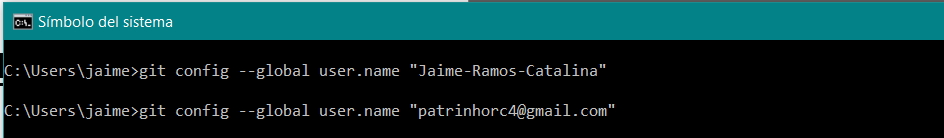
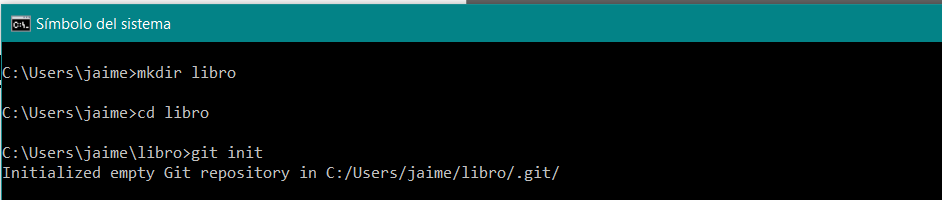
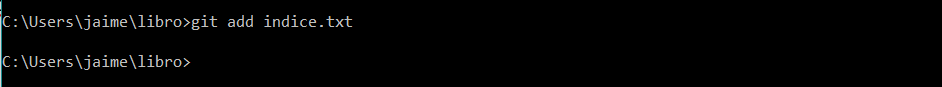
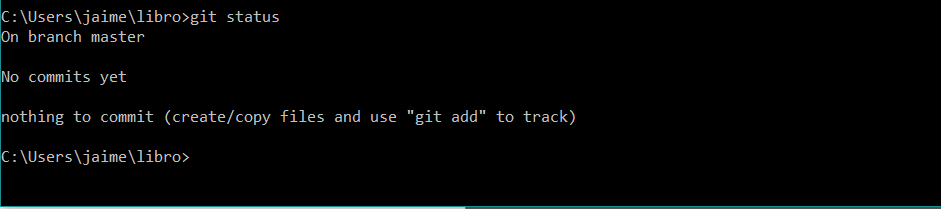
**SENTENCIAS UTILIZADAS GIT**

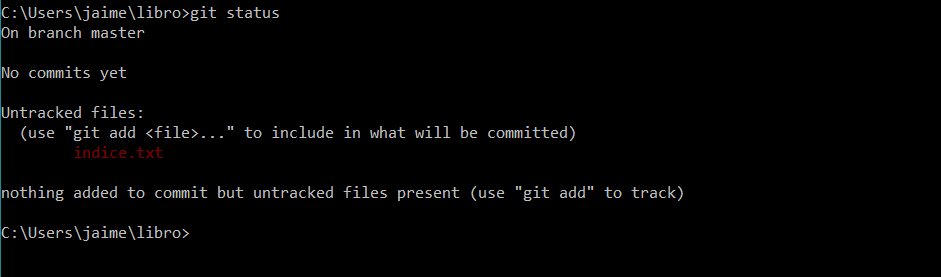
**CREATE:**

* **Git config:** Utilizamos la sentencia git config para configurar los datos de nuestro usuario de git. 
* **Git init:** Se utiliza una sola vez durante la configuración inicial de un repositorio nuevo. Al ejecutar este comando, se creará un nuevo subdirectorio . git en tu directorio de trabajo actual
* **Git add:** El comando git add añade un cambio del directorio de trabajo en el entorno de ensayo. De este modo, indica a Git que quieres incluir actualizaciones en un archivo concreto en la próxima confirmación. Sin embargo, git add no afecta al repositorio de manera significativa: en realidad, los cambios no se registran hasta que ejecutas git commit. 
* **Git clone:** El clon de Git se usa para copiar un repositorio de Git existente en un nuevo directorio local.

**SHOW:**

* **Git status:** El comando git status muestra el estado del directorio de trabajo. Permite ver los cambios que se han preparado, los que no y los archivos en los que Git no va a realizar el seguimiento





* **Git diff:**  es un comando multiusos de **Git** que, cuando se ejecuta, lleva a cabo una función para establecer las diferencias en los orígenes de datos de **Git**. Dichos orígenes de datos pueden ser confirmaciones, ramas y archivos, entre otras posibilidades.
* **Git diff <ID1> <ID2>:** Las difencias que hay entre ID1 e ID2.
* **Git log:** El comando git log muestra una lista de todas las confirmaciones realizadas en un repositorio.
* **Git blame:**es una utilidad de resolución de problemas versátil que cuenta con numerosas opciones de uso
* **Git show:** es una herramienta de línea de comandos que se utiliza para ver detalles ampliados en objetos de Git

**REVERT:**

* **Git reset:**  se utiliza para mover el proyecto a un commit anterior eliminando todos los posteriores de el historial de commits.
* **Git commit:** puedes recordar los cambios a los que se les hizo commits en una fecha posterior, o revertir el proyecto a esa versión
* **Git revert:** es una operación para deshacer de forma progresiva que ofrece una forma segura de deshacer los cambios
* **Git checkout:** te permite desplazarte entre las ramas creadas por git branch.

**UPDATE:**

* **Git fetch:** es un comando principal que se usa para descargar contenidos desde un repositorio remoto.
* **Git pull:**se emplea para extraer y descargar contenido desde un repositorio remoto y actualizar al instante el repositorio local para reflejar ese contenido.

**PUBLICH:**

* **Git commit -a:** Le dice al comando que organice automáticamente los archivos que se han modificado y eliminado, pero los archivos nuevos que no le ha informado a Git no se ven afectados.
* **Git push:** te permite subir los commits desde tu rama (branch) local en tu repositorio git local al repositorio remoto.