Simulador de Git — uGit

Este proyecto consiste en el desarrollo de un **simulador de sistema de control de versiones**, similar a Git, que permite a los usuarios realizar operaciones básicas como:

- Agregar archivos al área de staging
- Crear commits con mensajes
- Visualizar el historial de versiones
- Navegar entre versiones mediante checkout

Profesor

- Christian Vasquez

Integrantes

- Nicolás Álvarez
- Sebastián Vera
- Christofer Gutiérrez

Estructura del proyecto

incs/ — Archivos de cabecera (.h)

- repository.h — Define la estructura del repositorio, incluyendo el área de staging, HEAD y el índice hash de commits.
- staging.h — Define la estructura del área de preparación (StagingArea) y funciones para agregar, listar y liberar archivos.
- commit.h — Define la estructura de Commit, funciones para crear commits, listar historial, hacer checkout y liberar memoria.
- file.h — Define la estructura de archivos simulados, incluyendo nombre, contenido y tamaño.
- hashtable.h — Define la estructura de la tabla hash para indexar commits por ID y funciones de inserción, búsqueda y eliminación. src/

src/ — Archivos fuente (.c)

- main.c — Punto de entrada del programa. Procesa comandos del usuario, muestra el prompt, y coordina las operaciones.
- repository.c — Implementa la inicialización y liberación del repositorio, incluyendo el índice hash.
- staging.c — Maneja el área de staging: agregar archivos, listar y liberar.
- commit.c — Crea commits desde el staging, gestiona historial, búsqueda por ID y cambio de HEAD.
- file.c — Simula archivos con contenido ficticio y gestiona su memoria.

- hashtable.c — Implementa una tabla hash para búsqueda eficiente de commits por ID.

Otros

- Makefile — Script para compilar el proyecto fácilmente con make.
- README.md — Documentación del proyecto, estructura, comandos y guía de uso.

Comandos Implementados

- init -> Inicializa el repositorio
- add -> Agrega un archivo al staging
- commit -m "mensaje" -> Crea un commit con los archivos en staging
- log / history -> Muestra el historial de commits
- checkout -> Cambia HEAD al commit especificado
- help -> Muestra los comandos disponibles
- exit -> Sale del programa

Entorno de Desarrollo

- Sistema operativo: Windows 11
- Editor: Visual Studio Code
- Compilador: MSYS2 MINGW64 (gcc)

Ejecución desde Visual Studio Code

- Abrir la terminal integrada
- Verificar que el compilador esté instalado con gcc --version
- **Compilar el proyecto con**: gcc -Iincs -Wall -g src/main.c src/repository.c src/staging.c src/file.c src/commit.c src/hashtable.c -o ugit.exe
- Ejecutar el programa: ./ugit.exe
- Una vez finalizado el uso, eliminar el archivo ugit.exe

Ejecución desde MSYS2 MINGW64

- Abrir la terminal de MSYS2 MINGW64
- Navegar al directorio del proyecto
- Ejecutar el comando make para compilar usando el Makefile
- Iniciar con programa con ./ugit.exe
- Usar los comandos del simulador (init, add, commit, etc.)
- Al finalizar, limpiar los archivos generados con: make clean

Notas Técnicas

- El sistema utiliza una tabla hash para indexar commits por ID, permitiendo búsquedas en tiempo constante (O(1)).

- Los commits se enlazan en una lista simple, y cada uno contiene una copia profunda de los archivos simulados.
- El contenido de los archivos es ficticio, pero simula una estructura real con nombre, contenido y tamaño.
- El área de staging permite agregar hasta 1000 archivos antes de realizar un commit.